(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

The Development of an Intelligent Graphics Interface for the RESA Wargaming Simulation Terminals; A Proof of Concept

by

George Lee Yungk

June 1988

Thesis Advisor: J. S. Stewart II

Approved for public release; distribution is unlimited.

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable): Code 39	7a. NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
THE DEVELOPMENT OF AN INTELLIGENT GRAPHICS INTERFACE FOR THE RESA WARGAMING SIMULATION TERMINALS: A PROOF OF CONCEPT

12. PERSONAL AUTHOR(S)
Yunck, George L.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM _____ TO _____	1988 JUNE	140

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the DOD or the U.S. Government.

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Visual Interface, RESA, ATARI, Computer Graphics, Wargame Simulation, Computerized simulation; input output devices; (a)

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Comparisons have been made between many different methods of command input for wargame simulations. Much has been said and written about the relative merits of using a visual interface, menu and "mouse" input method for command input to various wargaming simulations. This document, which is the actual command interface program as implemented on an ATARI ST desktop computer, is Proof of the Concept that a "visual interface" as applied to the Research Evaluation Systems Analysis (RESA) simulation, is possible, given the complex command structure of RESA. (expanded)

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL CDR J.S. Stewart	22b. TELEPHONE (Include Area Code): 408-646-2419 22c. OFFICE SYMBOL: 55St

DD Form 1473, JUN 86 — Previous editions are obsolete. — SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

The Development of an Intelligent Graphics Interface for the RESA Wargaming Simulation Terminals; A Proof of Concept

by

George L. Yungk
Lieutenant Commander, United States Navy
B.S., University of Colorado, Boulder, 1975
M.S., University of Southern California, 1986

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY
(Space Systems Operations)

from the

NAVAL POSTGRADUATE SCHOOL
June 1988

Author: George L. Yungk
George L. Yungk

Approved by: J. S. Stewart II
J. S. Stewart II, Thesis Advisor

Linda K. Crumback
L. K. Crumback, Second Reader

R. Panholzer
R. Panholzer, Chairman
Space Systems Academic Group

G. E. Schacher
Gordon E. Schacher, Dean of
Science and Engineering

ABSTRACT

Comparisons have been made between many different methods of command input for wargame simulations. Much has been said and written about the relative merits of using a visual interface, menu and "mouse" input method for command input to various wargaming simulations. This document, which is the actual command interface program as implemented on an ATARI ST desktop computer, is Proof of the Concept that a "visual interface" as applied to the Research Evaluation Systems Analysis (RESA) simulation, is possible, given the complex command structure of RESA.

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND

The human interface to a computer program has always been very important and is the subject of countless books, studies and other documents. A good interface, one that is "user-friendly", has often been the major feature of a program, and its presence has often overcome the disadvantages of a mediocre, or in some cases, poor program. Additionally, what was a good or excellent interface in the past is now frequently archaic and is defined as poor or merely satisfactory. When terminology and devices such as graphics, color video, icons, windows, mouse input, voice recognition, light pens, touch screens, and intelligent terminals were unknown or considered exotic, typed text input predominated in input methodology for general use. Frequently, even when new interface technology became affordable, existing systems couldn't be adapted due to poor original program design or expense of conversion.

A poor interface is usually obvious to the user, and with perseverance, will be overcome. If the harm caused was only user frustration, little would be gained by trying to improve an interface. Unfortunately, a poor interface, while perhaps being exactly what a computer needs to perform its functions correctly, causes a user to perform inefficiently. This is

particularly significant because increased user (human) efficiency is the primary goal of most computer programs.

User inefficiencies, while not desirable, can be tolerated in many minor or simple applications. It's just not cost effective to spend large sums of money to create a sophisticated interface in those cases. But, if an interface is not user-friendly, or to be more precise, "user-efficient", the more complex subjects and applications will cause disproportionately greater user inefficiencies, making it worthwhile to find new ways to enhance the interface.

B. APPLICATION

The subject application of this thesis, Research Evaluation Systems Analysis (RESA) simulation, is a wargaming simulation utilized by the Naval Postgraduate School to do research on battle procedures and techniques, weapons use and wargame simulation itself. RESA operates on the Digital Equipment Corporation VAX 11 series minicomputer. User input (game commands) is typed text entry, often in response to program prompts at VT-100/102 "dumb" terminals. RAMTEK graphics terminals provide geo-graphic color displays. A typical user is a graduate student who has no prior knowledge of RESA, and, probably little, or no experience in any form of wargaming or computer simulation. Additionally, few have any significant experience in interfacing with mainframe computers or sophisticated mainframe programs.

Because of the limited time a typical team of users will actually be using RESA, efficiency in interfacing is extremely critical. Several factors militate against this desired efficiency.

1. Text Entry Requirement

All input, no matter how simple or trivial, must be typed on a keyboard. Spelling must be exactly correct. Since user spelling and typing abilities vary widely, the progress of a simulation exercise is often dictated by the input speed of a lesser skilled team member. Spelling help is available on the input terminal, but frequently requires a diversion from the command input sequence.

2. Command Syntax Knowledge Needed

While RESA commands are one word, and are usually descriptive of the action being directed, they must be entered in the correct context. Additionally, sequences of commands must frequently be used to accomplish certain actions, and the sequence must be exactly correct. These sequences do not always lend themselves to easy memorization, thus requiring written user guides, or constant diversions from the command input sequence to get limited on-line help.

3. Separate Terminals For Different Functions

Currently, the users of RESA require a minimum of three separate terminals to conduct the simulation. One terminal is for user control of the game which provides for command input and computer response. Another terminal is

necessary for display of the various status boards which give current game and unit (ships, planes, etc) data. A third terminal is used for display of color graphics which provide a birds-eye view of the current scenario. If a user wants to be informed about a facet of the simulation or enter a command, he must physically go to the appropriate terminal and interrupt the user at that terminal to conduct the desired activity.

From examination of the above factors, it is evident that users spend a considerable amount of time on the wargame input process, time that should be spent on the simulation strategy. It is to improve this interface, thereby increasing user efficiency, that enhanced input methodology is constantly being studied.

II. SYSTEM REQUIREMENTS

A. HARDWARE

In selecting the hardware to prove the concept of a visual interface as a more efficient method to control RESA, several criterion were considered critical to success of the Proof of Concept.

1. Visual Interface Capability

The system selected would have to be capable of iconographics, menus, and other graphics displays. Input devices such as the mouse, trackball, joystick, and touch tablet, as well as the standard keyboard, must be supported. The implication, but not absolute requirement, of this required capability is that the system must support "event-driven" programming. "Event-driven" is the term used to describe the computer's (both hardware and software) method of handling other than typed text input. It is most often used in reference to mouse/menu capabilities, as implemented on the ATARI ST and Macintosh computers.

2. Advanced Technology

While some older systems are able to do much of what is described in A.1. above, the visual interface is done by "brute force" and is frequently stretching the limits of that system's capability. State-of-the-art technology means that the system has been designed, and therefore optimized,

for the display requirements stated above. An advanced technology system would also provide for expansion more easily than a system already dated.

3. Memory

Graphics require large amounts of memory and high machine speed to produce good displays. A minimum of one megabyte of memory would ensure that, for this application, memory limitations were not a factor. Future system expansion would also be facilitated.

4. Color Display

For screen displays that are mostly text, color would not be necessary, but would greatly enhance the interface. To allow creation of 100% graphic displays, such as a geographic battle group display, color would be necessary.

5. Cost

Finding systems with the above attributes isn't particularly difficult; the issue is one of money. The system selected must be inexpensive enough to replace the existing "dumb" terminals at a reasonable cost. Studies to evaluate system cost versus user efficiency are not part of this document. Relative system cost is the selection criteria used here.

B. SOFTWARE

The software criteria, while much less critical than the hardware criteria, must still meet certain standards to enable this Proof of Concept to be successful.

1. Access To Hardware Potential

Software must be capable of using all hardware capabilities. Emphasis will be on graphics and color, not numeric calculations.

2. Ease Of Use

Many current languages are capable enough to do almost anything asked of them, if enough time and skilled programmers are available. For this Proof of Concept, time is limited, and professional programmers will not be available.

3. Speed

Even if hardware has been selected to maximize graphic displays, the software must execute quickly to fully use that capability.

4. Cost

The software must be chosen using criteria that provide the above software capabilities as inexpensively as possible.

C. THE DECISION

The ST series of computers by the ATARI Corporation was chosen to meet the above hardware needs. Various versions are available in 500Kb to 4Mb versions, use Motorola's 68000 chip (16-bit) technology, use Digital Research's GEM icon / menu / mouse operating interface, have high resolution color

graphics, are readily available across the U.S., and are the least expensive systems, by far, that meet the stated criteria.

The software chosen was GFA BASIC, an inexpensive BASIC very similar to TURBO-BASIC in the MS-DOS world. It is very easy to use, accesses all machine capabilities, and when compiled, is faster than PASCAL, and rivals the "C" language in speed of execution.

III. THE CONCEPT

A. THE PROBLEM

Interaction with the RESA simulation using the current methodology is quite inefficient. Commands, or sequences of commands called orders, are typed in using a keyboard at a dedicated "dumb" input terminal. For most typical users, this means frequent time-consuming interruptions to obtain absolutely correct commands, call signs, weapons names, flight and track data, etc., either from written user guides, or from a second terminal. In addition to simple spelling mistakes, syntax errors are frequently made when forming orders.

In order to keep a using team abreast of the scenario, a second (dumb) terminal must be configured as a status board terminal, is not available as an input terminal, and requires an additional player.

A third position must be used if a non-textual view of the situation is desired. Two Ramtek color graphics monitors provide Geotactical Displays which give the user a graphical representation of the RESA simulation from an overhead perspective, much like a radar PPI display. Unfortunately, the Geotactical Display is not controlled at the color monitor itself, but at a physically separate (text) input terminal. Only one representation at a time is normally

available to each of the four user teams. These four graphics processes make a substantial burden on the single VAX computer which hosts the simulation. Since it is a fairly slow process to change and redisplay the Geotactical Display, compromises are inevitably made in selecting the scale or symbology for display. It is unarguable that the graphical representation best suited to prosecute the outer air battle is not the one best suited for battle group inner missile defense.

The time loss and distractions that result from use of the current interface detract considerably from the user's attention to the actual battle simulation decisions and strategy.

B. THE PROGRESS

This document is a Proof of Concept of a visual interface, menu and "mouse" input method for the RESA simulation, and is a logical progression of earlier interface studies. The following is a short synopsis of the progression of studies leading to this thesis:

[Manson, 1985]--Conducted experiment to compare speech and keyboard inputs to Naval Warfare Interactive Simulation System (NWISS, predecessor to RESA) in adverse conditions of lighting and noise. While spelling and typing problems may be solved by this interface method, the other input factors remain unsolved. Additionally, speech recognition input adds

the complication of inaccuracies in input due to current equipment limitations.

[Irving, 1986]--Project to use A Macintosh microcomputer as a command input terminal for NWISS. Use was made of the "windowing" and menu/mouse selection methodology to avoid command syntax errors and speed command entry.

[Sweeney, 1986]--Comparison of a "visual" Macintosh interface and voice command input to the standard keyboard entry method was made. Continuous voice input was favored over the visual interface if training time was not a significant restriction.

[LeFever, 1987]--Investigated further the use of voice input methods to RESA, using the VOTAN continuous speech recognizer. Explored the complications arising from the need to categorize the game commands, and the inability to establish a tree architecture for correct command structure.

[Lower, 1987]--Examined enhancement of player input to the Joint Theater-Level Simulation (JTLS) by using a visual interface method implemented on a desktop microcomputer, an Apple Macintosh. Showed the feasibility of coding of a prototype interface (in Pascal), and laid out a sample program skeletal structure. The input process was streamlined somewhat, but coding was difficult and incomplete.

[Adams, 1987]--Demonstrated the use of graphics and screen menus for the display of command and control information on a dedicated color graphics workstation, the IRIS. User input to control a simulation was not addressed here, but display concepts were evolutionary and menus as item selectors were effectively used.

[Copeland, 1987]--Pursued the concept of the visual interface as primary input methodology. Windowing and menu/mouse usage, as normally implemented on a Macintosh, was applied to the JTLS and Battle Group Tactical Trainer (BGTT) user interfaces. The framework for a generic architecture for this type of interface was discussed. Desirable features of a visual interface were also discussed.

[Stevens, 1987]--Demonstrated the feasibility of improving and enhancing the user interface of the BGTT by developing a visual interface prototype. Code for a small subset of the user I/O process was written. MACTRAN 77, a Macintosh version of FORTRAN 77, was used as the language. A communications driver was not made available, so actual interface with RESA on the VAX was not accomplished.

All previous programming efforts resulted in incomplete work which was innovative, but not a full implementation of the visual concept. This Proof of Concept effort is an extensive and complete program, which, when interfaced to the RESA host computer, promises a fast and efficient interface capability not previously achieved.

C. THE PLAN

A full-featured, efficient interface for RESA would serve several functions, and should be developed in several logical phases. Each of the following functions is currently being done, but it takes at least three different terminals to accomplish it. A complete "smart" interface program would handle all of these functions from a single terminal, with immediate access to each function through the menu system.

1. System Operation--The ability for system operators to set up and start RESA, and for "umpires" to control simulation events.

2. Command Input--The ability to quickly create error-free command strings.

3. Communication--The ability to allow actual data flow between the user and RESA.

4. Status Boards--The ability to display the currently generated data from the RESA simulation.

5. Geo-graphics--The ability to display the current scenario and/or status of units of the battle group in color graphics.

Time and scope limit this Proof of Concept to Phases 2 and 3, with some work on 1.

IV. THE RESA INTERFACE PROGRAM (RIP)

A. OVERVIEW

The RIP was designed to be a smart interface between the user and RESA. It operates entirely on the ATARI ST as a background program, and can not be distinguished by the VAX as other than a "dumb" terminal. When implemented as a full-featured program, the system operator, during boot-up, will be able to easily configure (Figure 1) the terminal according to different users' skill levels. No user access to the program code is required or desireable. The command sequences, or orders, are created in their entirety on the ST and then sent to the VAX.

Although their use was not precluded, no attempt to use the RESA error-checking or on-line help capabilities was made, as this would unnecessarily complicate the RIP. The inherent characteristics of the ST's drop-down menu system make on-line help largely redundant, and keyboard entry error-checking was handled by the RIP.

B. PROGRAM OPERATION

In the current version of the program, the operator is presented an interface which runs rapidly, provides error checking and a logical sequencing of the events necessary to build any required order and send it as an error-free product. The code runs very swiftly such that no delay is

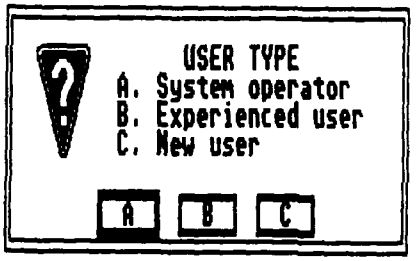


Figure 1

Configuration display screen.

generated by the computer interface. Capabilities of the ATARI have been used to present windows and boxes which remain until a selection has been made. Less hand motion is required on the mouse than with previous attempts on the Macintosh. The user is free to move from mode to mode at any time unless such a move would be counter-productive, in which case it is prohibited, and audio cues are generated when inadvisable attempts are made. A correct degree of automation has been attained considering the requirement for alpha-numeric input variability which is high during game play and generally unknown in scope at game outset. In addition, the work surface is attractive and easy to see.

The user sits at a terminal where the RIP is running. RESA system operators would normally be the ones to have started the system and activated each terminal, but RIP initialization on the ST is simple enough for ordinary users.

The starting screen display (Figure 2) will provide the Main Menu, one of the three primary control menus. The other two are: Force Menu A and Force Menu B. Access to each primary menu is available from each other primary menu using the "new Menu" menu bar selection. Figure 3 shows the "drop-down" menu choices for "new Menu" available during a Main Menu screen display.

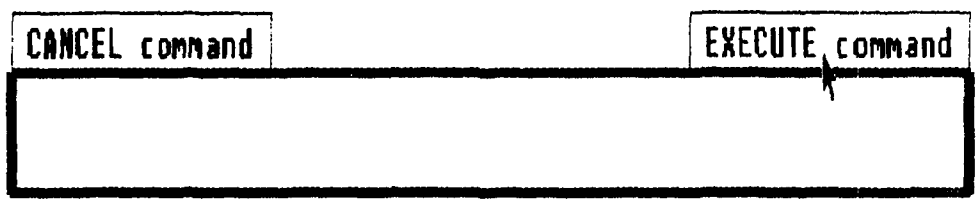


Figure 2

Starting display screen.

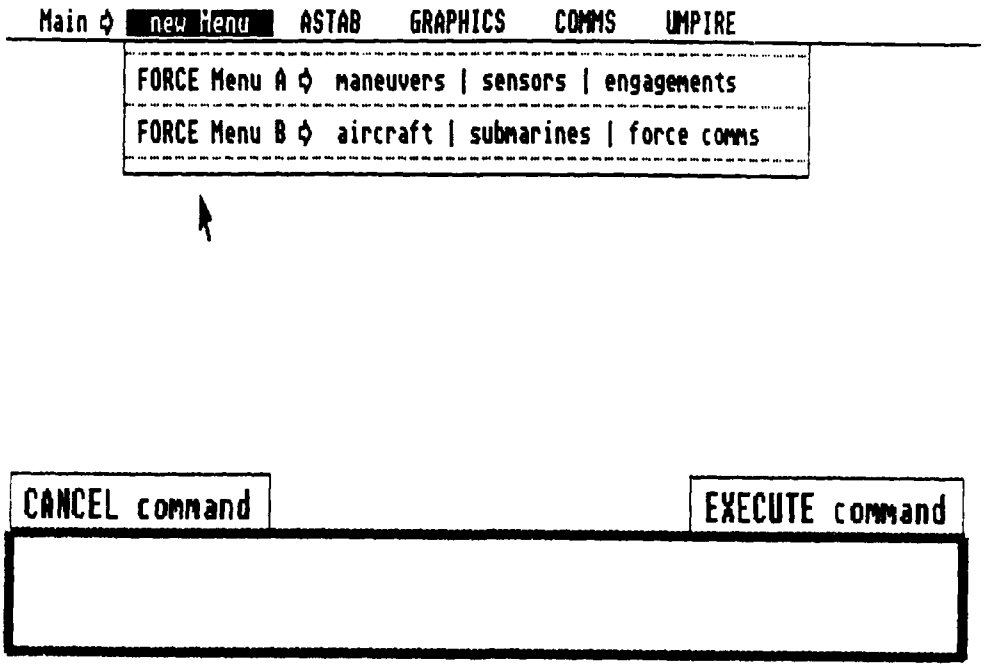


Figure 3

Primary menu selection screen.

Orders are "built" by successively selecting menu headings and using the "mouse" to select desired commands. As commands are selected, the RIP branches programmatically to ask for necessary data or additional commands. Users are directed to make a specific choice from the screen or a secondary menu, or to use the keyboard to enter alpha-numeric characters. Considerable error-checking is performed if the keyboard is used. As they are being built, orders are displayed in a command box at the bottom of the screen as shown in Figure 4. When an order is syntactically correct, the user is given the choice to Execute or Cancel it.

Execution of partial orders is not allowed, but Cancellation at any time is possible by pressing the Control / Shift / Alternate keys simultaneously.

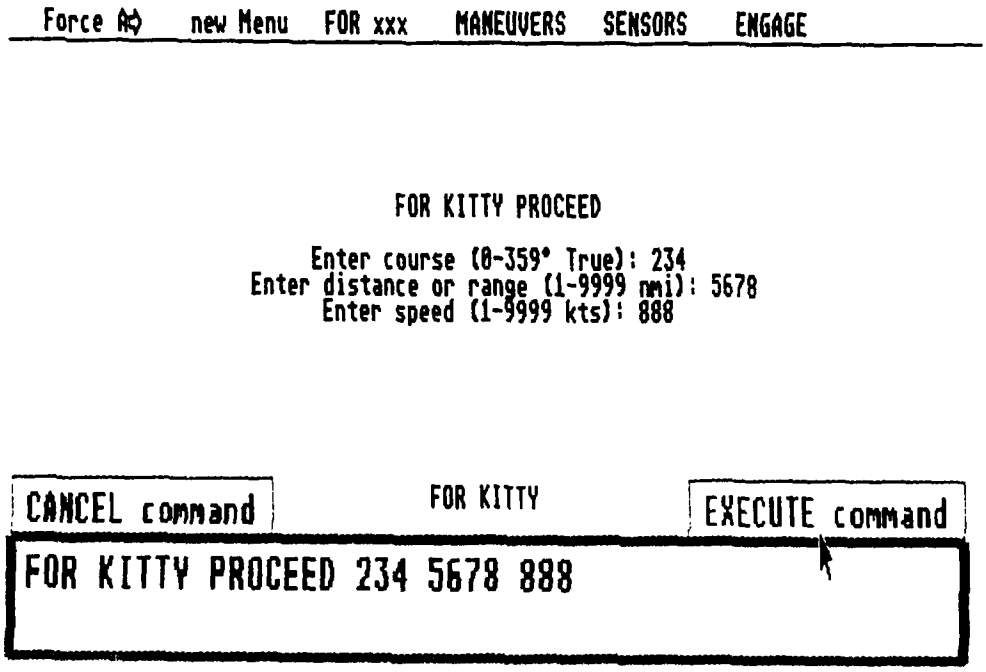


Figure 4

Example of commands built to form an order.

C. THE CODE

The BASIC language allows programmers nearly unlimited latitude in program structure, and when programmer discipline is minimal, BASIC programs frequently look more like free-form art than functional software. To allow others to easily understand and enhance the RIP, a rigid structure (Figure 5) and logical program flow was used throughout. The program is segmented into mainly short, similar procedures. Repetitive functions are contained in common procedures whenever possible. The RIP Code is included as Appendix A.

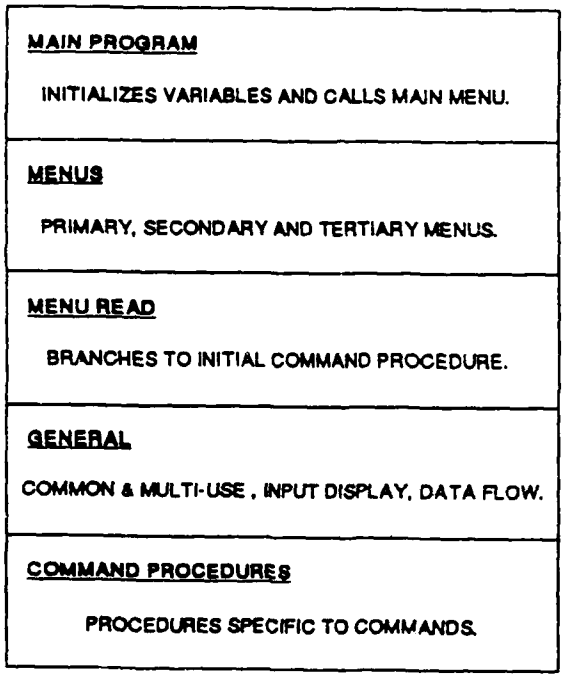


Figure 5

RESA Interface Program Structure.

Generally, program flow (Figure 6) is as follows:

1. Display a primary menu.
2. Wait for menu selection.
3. Branch to a menu_read procedure.
4. Branch to procedure for specific command.
5. Follow command "tree" until order is complete.
6. Wait to Execute or Cancel order.
7. Display a primary menu (back to step 1).

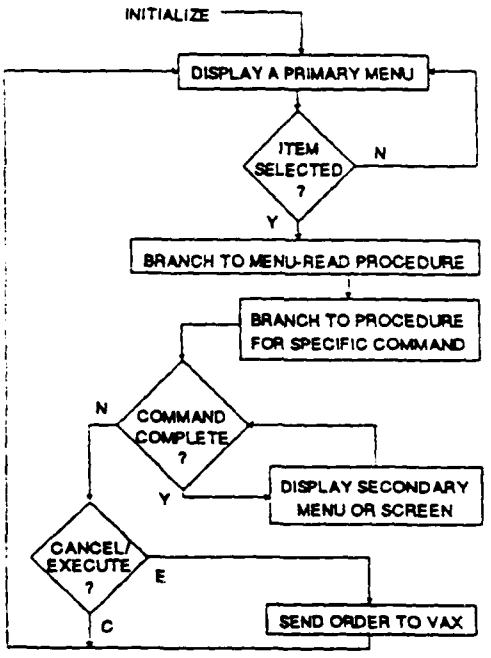


Figure 6

RESA Interface Program Flowchart

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The concept to be proven was that an inexpensive color graphics microcomputer could serve as an "intelligent" terminal for the RESA wargaming simulation. Success depended on whether the command structure of RESA could be effectively represented in code and implemented on a system meeting the aforementioned requirements. The operating program (RIP) that has been created generates such an interface which operates rapidly, and can incorporate the entire command structure for RESA with room for future expansion. In addition, the chosen hardware/software combination provides a windowing environment which can be used to support all RESA player functions on one very cost-effective terminal.

The program runs very efficiently and demonstrates many visual interface criteria developed by previous researchers as well as improving on many previous suggestions. The concept has been shown to be not only practical, but available for use and robust where future changes are concerned. While extensive user testing has not been attempted, and many enhancements remain unfinished, preliminary indications are that considerable user efficiency will be gained by use of the user interface developed in this paper.

B. RECOMMENDATIONS FOR FURTHER STUDY

In order to bring the process started here to its optimal conclusion, and allow the RIP to produce the kind of efficiencies that are possible, several additional steps must be taken.

1. Accomplish the data interface with the VAX/RESA system and ensure VAX-generated messages are automatically displayed in "attention" windows.

2. Conduct studies of the actual efficiency gain from using the RIP, or portions of it.

3. Conduct user experiments to fully optimize RIP displays and interactive usage.

Having accomplished the interface, determine the techniques required and code changes to route four types of RESA information to the same serial port. Then enhance the RIP in the following fashion.

First, write code for display of the ASTABS. Each one could be called from a menu and displayed in a separate window while still sending orders to RESA. It is even quite possible to use the mouse cursor to point directly at ASTAB items for input in the command structure, thereby reducing the slower keyboard usage even further. New status displays could be developed and made a permanent part of the RIP, or provisions could be made to allow each user to create his own. Test the code and integrate into the RIP.

Write code for display of Geo-graphics. Steps to enhance the graphics display would be similar to those described for the ASTABS. Additionally, unit movement or positioning commands could be done as simply as pointing at the unit and pointing where you wanted it to go. Features like those in microcomputer "paint & draw" programs could be included to allow for initial positioning of forces. Test the code and integrate into the RIP.

Write code to create different interfaces for different skill levels of users, i.e., System Operator, Umpire, Experienced User, Novice User. This would allow only the commands that the particular user needed to be accessible, or various levels of Help to be automatically provided. Test the code and integrate into the RIP.

Add peripheral enhancements. Utility programs can be installed during boot-up of the ATARI ST, and used by simply touching the menu bar during RIP operation. A user could have instant access to a calculator, notepad, references on ships or planes, and/or other useful functions.

Finally, explore the translation of the RIP to the Enhanced Naval Warfare Gaming System (ENWGS) which has been selected to replace the Navy's primary wargaming system. Use of this concept would greatly reduce the effort needed to operate the ENWGS simulation from each of the many terminals currently required, and reduce the cost of system requirements.

APPENDIX A

```
'                    RESA Interface Program Code (RIP)
' ##############################################################################
' ##############################################################################
'
'      MAIN PROGRAM
'
' ##############################################################################
'
' On Break Cont  ! deactivates "break" capability.
@Init           ! for initializing & dimensioning
@User_type      ! allows choice of type of user; NOT YET WRITTEN !
@Save_blanks    ! saves blank screen areas to use to clear screen after inputs
@Draw_box       ! draws box for output string
@Main_menu      ! main menu
'
' ##############################################################################
'
'      MENU PROCEDURES
'
' ##############################################################################
'
'  Produces dialog box to enter "type" of user; WHEN COMPLETE, this could
'        allow various levels of access to the system commands.
'
Procedure User_type
  Cstr$="Welcome to the RESA Interface Program."
  Cstr2$="Naval Postgraduate School"
  Deftext 1,0,0,13
  Text 120,50,400,Cstr$
  Deftext 1,1,0,13
  Text 120,170,400,Cstr2$
  Deftext 1,0,0,6          !## resets text type to normal
  Mtxt$="     USER TYPE A. System operator B. Experienced user C. New user"
  Alert 2,Mtxt$,1," A   B   C ",A
  Print A
  Clr A
Return   !@User_type
'
'                                                                  MAIN Menu
Procedure Main_menu
  @Cclear_middle
  Void Fre(0)                  ! clean up variables
  Restore Mmain_data
  For I=0 To 90                ! set up a loop
    Read Bar$(I)               ! read data from data field
    Exit If Bar$(I)="***"      ! until end of data field
  Next I
```

```
  Bar$(I)=""                ! tail blanks into string
  Bar$(I+1)=""              ! ditto
  Mmain_data:
  Data Main ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  '
  Data  new Menu ,-----------------------------------------------------
  Data  FORCE Menu A ¬C  maneuvers   sensors   engagements
  Data ---------------------------------------------------
  Data  FORCE Menu B ¬C  aircraft   submarines   force comms
  Data ---------------------------------------------------,""
  '
  Data  ASTAB ,-, Bearing, Classify, CPA, Designate , Drop, Print, Show,""
  '
  Data  GRAPHICS ,-, Plot, Erase, Center, Radius, Shift, Label, LOB
  Data  Mark track, Mark bearing, Unmark track, Unmark bearing
  Data  Place, Cancel,""
  '
  Data  COMMS ,-, Inform, Intell, Message ,""
  '
  Data  UMPIRE ,-, Go, Pause, End
  Data  Copy, Relocate, Save, Time, Set
  Data  Enable,  Disable, Expend,  Replenish,""
  Data ***
  '
  Menu Bar$()                       ! activate menu
  On Menu Key Gosub Help_key_test
  On Menu  Gosub Main_menu_read
  On Menu Ibox 1,450,142,180,18 Gosub Inbox_execute
  On Menu Ibox 2,5,142,170,18 Gosub Inbox_cancel
  Do
    On Menu
  Loop
Return   !@Main_menu
'
'                                                                FORCE Menu A
Procedure Force_menu_a
  @Cclear_middle
  Void Fre(0)             ! clean up variables
  Firsttime!=True         ! allows "Weapons Tight/Free" to register only once
  Restore Fforce_a_data
  For I=0 To 110            ! set up a loop
    Read Bar$(I)            ! read data from data field
    Exit If Bar$(I)="***"   ! until end of data field
  Next I
  Bar$(I)=""                ! tail blanks into string
  Bar$(I+1)=""              ! ditto
  Fforce_a_data:
  Data Force A¬C,  ??
  Data --------
  Data 1,2,3,4,5,6,""
```

```
  Data   new Menu ,-----------------------------------------------------
  Data  MAIN Menu ¬C  astab   graphics   player comms   game
  Data ---------------------------------------------------
  Data  FORCE Menu B ¬C  aircraft   submarines   force comms
  Data ----------------------------------------------------,""
  '
  Data  FOR xxx ,-, Select unit ,""
  '
  Data  MANEUVERS ,-, Course, Speed, Proceed, Station
  Data  Search, USE (plan) , Execute (plan)
  Data  Enter Orders, Pending Orders , Cancel,""
  '
  Data  SENSORS ,-, Activate , Silence, Blip on,  Blip off , DECM on
  Data   DECM off , RBOC on,  RBOC off , Jam, Cease, Emcon,""
  '
  Data  ENGAGE ,-, Weapons , Fire, Launch, Take,""
  Data ***
  '
  Menu Bar$()                        ! activate menu
  On Menu  Gosub Force_menu_a_read
  On Menu Ibox 1,450,142,180,18 Gosub Inbox_execute
  On Menu Ibox 2,5,142,170,18 Gosub Inbox_cancel
  Do
    On Menu
  Loop
Return   !@Force_menu_A
'
'                                                          FORCE Menu B
Procedure Force_menu_b
  @Cclear_middle
  Void Fre(0)                ! clean up variables
  Restore Fforce_b_data
  For I=0 To 110  !90             ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Fforce_b_data:
  Data Force B¬C,  ??
  Data --------
  Data 1,2,3,4,5,6,""
  Data   new Menu ,-----------------------------------------------------
  Data  MAIN Menu ¬C  astab   graphics   player comms   game
  Data ---------------------------------------------------
  Data  FORCE Menu A ¬C  maneuvers   sensors   engagements
  Data ----------------------------------------------------,""
  '
  Data  FOR xxx ,-, Select unit ,""
  '
  Data  AIRCRFT ,-, Launch ,-, Flight Cmds ,-, Alert, Close
  Data  Handover, Open, Orbit, Recall, Recover,""
```

```
  '
  Data  SUBMRINE ,-, Depth, Surface, Periscope , Fire
  Data  Mode, Mast, Deploy, Retrieve,""
  '
  Data  COMMS ,-, Commtext , Embark, Report, Circuit,""
  Data ***
  '
  Menu Bar$()                        ! activate menu
  On Menu  Gosub Force_menu_b_read
  On Menu Ibox 1,450,142,180,18 Gosub Inbox_execute
  On Menu Ibox 2,5,142,170,18 Gosub Inbox_cancel
  Do
    On Menu
  Loop
Return   !@Force_menu_B
'
'                                                          Designate sub-menu
Procedure Designate_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Ddesignate_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Ddesignate_data:
  Data Desig ¬C ,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data  as... ,-, Enemy, Friendly , Neutral, Unknown,""
  Data ***
  Cstr$="select Designation..."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                    ! activate menu
  On Menu  Gosub Designate_menu_read
  Do
    On Menu
  Loop
Return   !@Designate_menu
'
'                                                          Show sub-menu
Procedure Show_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Sshow_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Sshow_data:
```

```
  Data  Show ¬C ,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  '
  Data  A - C  ,-, AAWC, Active, Air, ASUWC, ASWC, Bogey (tote & cap)
  Data  Continuation (of next page) ,""
  '
  Data  D - P  ,-, Damage (& reconn info) , ESM, EWC, Flight, Force, HFDF
  Data  Intell (spot reports), Passive (sonar tracks),""
  '
  Data  Q - Z  ,-, Reporting (policies), Ship, Shore, SOSUS (tracks)
  Data  Submarine, Surface (tracks), Surveillance (satellites) , Weather,""
  Data ***
  '
  Cstr$="Select item to Show status information for..."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()
  On Menu  Gosub Show_menu_read
  Do
    On Menu
  Loop
Return   !@Show_menu
'
'                                          Display sub-sub-menu (of Show menu)
Procedure Display_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Ddisplay_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Ddisplay_data:
  Data Display¬C ,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data   on... ,-, Blue, Neutral, Orange, <astab>, <continue> ,""
  Data ***
  Cstr$="Select which display to show information on."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Display_menu_read
  Do
    On Menu
  Loop
Return   !@Display_menu
'
'                                          Show Air sub-sub-menu (of Show menu)
Procedure Show_air_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Showair_data
```

```
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Showair_data:
  Data  Show ¬C ,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data   air... ,-, Alert, Availability , Events, Tracks,""
  Data ***
  Cstr$="Select AIR item."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Show_air_menu_read
  Do
    On Menu
  Loop
Return   !@Show_air_menu
'
'                                                          Plot sub-menu
Procedure Plot_menu
  Menu Kill
  Restore Pplot_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Pplot_data:
  Data Plot ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data  new Menu ,-------------------------------------------------------
  Data  MAIN Menu ¬C  astab   graphics   player comms   game
  Data ---------------------------------------------------
  Data  FORCE Menu A ¬C  maneuvers   sensors   engagements
  Data ---------------------------------------------------
  Data  FORCE Menu B ¬C  aircraft   submarines   force comms
  Data ---------------------------------------------------,""
  '
  Data    PLOT     ,-, All, Blue, Orange, Own, Boundaries , Chaff, LOB
  Data  Regions, Rivers, Sonobuoy, Speed, Survsat, PIM, Track, Station,""
  '
  Data ***
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Plot_erase_menu_read
  Do
    On Menu
    T$="PLOT "    ! ensures "PLOT" precedes each Plot command.
```

```
  Loop
Return   !@Plot_menu
'
'                                                          Erase sub-menu
Procedure Erase_menu
  Menu Kill
  Restore Eerase_data
  For I=0 To 90                ! set up a loop
    Read Bar$(I)               ! read data from data field
    Exit If Bar$(I)="***"      ! until end of data field
  Next I
  Bar$(I)=""                   ! tail blanks into string
  Bar$(I+1)=""                 ! ditto
  Eerase_data:
  Data Erase ¬C,  ??
  Data -------
  Data 1,2,3,4,5,6,""
  Data  new Menu ,-----------------------------------------------------
  Data  MAIN Menu ¬C  astab   graphics   player comms   game
  Data -----------------------------------------------------
  Data  FORCE Menu A ¬C  maneuvers   sensors   engagements
  Data -----------------------------------------------------
  Data  FORCE Menu B ¬C  aircraft   submarines   force comms
  Data -----------------------------------------------------,""
  '
  Data   ERASE   ,-, All, Blue, Orange, Own, Boundaries , Chaff, LOB
  Data  Regions, Rivers, Sonobuoy, Speed, Survsat, PIM, Track, Station,""
  '
  Data ***
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Plot_erase_menu_read
  Do
    On Menu
    T$="ERASE "   ! ensures "ERASE" precedes each Erase command.
  Loop
Return   !@Erase_menu
'
'                                                          Cancel sub-menu
Procedure Cancel_menu
  Menu Kill
  Restore Ccancel_data
  For I=0 To 90                ! set up a loop
    Read Bar$(I)               ! read data from data field
    Exit If Bar$(I)="***"      ! until end of data field
  Next I
  Bar$(I)=""                   ! tail blanks into string
  Bar$(I+1)=""                 ! ditto
  Ccancel_data:
  Data Cancel ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data  new Menu ,-----------------------------------------------------
```

```
  Data  MAIN Menu ¬C  astab   graphics   player comms   game
  Data ----------------------------------------------------
  Data  FORCE Menu A ¬C  maneuvers   sensors   engagements
  Data ----------------------------------------------------
  Data  FORCE Menu B ¬C  aircraft   submarines   force comms
  Data ----------------------------------------------------,""
  '
  Data  A - B ,-, Activate , All, Altitude, Attach, Barrier, Bingo, Blip,""
  '
  Data  C - D ,-, Cease, Chaff, Circle , Course, Cover, DECM
  Data  Deploy, Depth, Detach,""
  '
  Data  E - M ,-, Emcon, Execute , Fire, Grid, Jam, Launch
  Data  Mast, Mission, Mode,""
  '
  Data  N - R ,-, Orbit, Proceed, RBOC, Recall, Reconn, Recover
  Data  Refuel, Retrieve ,""
  '
  Data  S - Z ,-, Search, Silence , Speed, Station, Take, Turn
  Data  Weapons, Xmark,""
  '
  Data ***
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Cancel_menu_read
  Do
    On Menu
    T$="CANCEL "   ! ensures "CANCEL" precedes each Cancel command.
  Loop
Return   !@Cancel_menu
'
'                                                          Weapons sub-menu
Procedure Weapons_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Wweapons_data
  For I=0 To 90             ! set up a loop
    Read Bar$(I)            ! read data from data field
    Exit If Bar$(I)="***"   ! until end of data field
  Next I
  Bar$(I)=""                ! tail blanks into string
  Bar$(I+1)=""              ! ditto
  Wweapons_data:
  Data Weapons¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data    FREE     ,-, Air, Surface, Submarine, All, Enemy
  Data  Nuclear, Conventional ,""
  Data   TIGHT     ,-, Air, Surface, Submarine, All, Enemy
  Data  Nuclear, Conventional ,""
  Data ***
  Cstr$="select Weapon item..."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                  ! activate menu
```

```
  On Menu  Gosub Weapons_menu_read
  Do
    On Menu
  Loop
Return   !@Weapons_menu
'
'                                                    Launch Cruise sub-menu
Procedure Cruise_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Ccruise_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Ccruise_data:
  Data Cruise ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data   mode  ,-, BOL, PL2, PL3, PLTWO, PLTHREE , TLAM,""
  Data ***
  Cstr$="select Cruise mode..."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                  ! activate menu
  On Menu  Gosub Cruise_menu_read
  Do
    On Menu
  Loop
Return   !@Cruise_menu
'
'                                                  SENSOR Activate sub-menu
Procedure Activate_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Aactivate_data
  For I=0 To 90              ! set up a loop
    Read Bar$(I)             ! read data from data field
    Exit If Bar$(I)="***"    ! until end of data field
  Next I
  Bar$(I)=""                 ! tail blanks into string
  Bar$(I+1)=""               ! ditto
  Aactivate_data:
  Data Sensor¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data  activate ,-, Air, Approach , Emitter, ESM, Radar, Sonar
  Data  Surface, Survsat,""
  Data ***
  Menu Bar$()                  ! activate menu
  Print At(31,Ytext%);"Select menu item..."
  '
  If Aclaunch!
```

```
    T$=T$+"ACTIVATE "
  Else
    T$=F_name$+" ACTIVATE " ! ensures "ACTIVATE" precedes each Activate cmd.
  Endif
  '
  On Menu  Gosub Activate_menu_read
  Do
    On Menu
  Loop
Return   !@Activate_menu
'
'                                                    Activate SONAR mode sub-menu
Procedure Sonar_menu
  @Cclear_middle
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Ssonar_data
  For I=0 To 90             ! set up a loop
    Read Bar$(I)            ! read data from data field
    Exit If Bar$(I)="***"   ! until end of data field
  Next I
  Bar$(I)=""                ! tail blanks into string
  Bar$(I+1)=""              ! ditto
  Ssonar_data:
  Data  Sonar ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data    mode... ,-, BB, CZ, DP, none ,""
  Data ***
  Menu Bar$()                  ! activate menu
  Cstr$="Select a mode."
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  On Menu  Gosub Sonar_menu_read
  Do
    On Menu
  Loop
Return   !@Sonar_menu
'
'                                                        SENSOR Silence sub-menu
Procedure Silence_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Ssilence_data
  For I=0 To 90             ! set up a loop
    Read Bar$(I)            ! read data from data field
    Exit If Bar$(I)="***"   ! until end of data field
  Next I
  Bar$(I)=""                ! tail blanks into string
  Bar$(I+1)=""              ! ditto
  Ssilence_data:
  Data Sensor¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data   silence ,-, Air, Approach , Emitter, ESM, Radar, Sonar
```

```
  Data  Surface, Survsat,""
  Data ***
  Menu Bar$()                  ! activate menu
  Print At(31,Ytext%);"Select menu item..."
  '
  If Aclaunch!
    T$=T$+"SILENCE "
  Else
    T$=F_name$+" SILENCE " ! ensures "SILENCE" precedes each Silence cmd.
  Endif
  '
  On Menu  Gosub Silence_menu_read
  Do
    On Menu
  Loop
Return   !@Silence_menu
'
'                                                    AIRCRAFT mission sub-menu
Procedure Mission_menu
  @Cclear_middle
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Mmission_data
  For I=0 To 90            ! set up a loop
    Read Bar$(I)           ! read data from data field
    Exit If Bar$(I)="***"  ! until end of data field
  Next I
  Bar$(I)=""               ! tail blanks into string
  Bar$(I+1)=""             ! ditto
  Mmission_data:
  Data Aircrft¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data   mission ,-, none, AEW, Airtanker, ASW, CAP, Decoy, EW, Jammer
  Data  Reconn, Relay, Rescue, Search, Strcap, Strike
  Data  Sttanker, Surcap, Surveillance ,""
  Data ***
  Menu Bar$()                  ! activate menu
  Cstr$="Select a mission"
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  On Menu  Gosub Mission_menu_read
  Do
    On Menu
  Loop
Return   !@Mission_menu
'
'                                                AIRCRAFT commands sub-menu
Procedure Flt_commands_menu
  @Cclear_middle
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Commands_data
  For I=0 To 90            ! set up a loop
    Read Bar$(I)           ! read data from data field
```

```
    Exit If Bar$(I)="***"  ! until end of data field
  Next I
  Bar$(I)=""               ! tail blanks into string
  Bar$(I+1)=""             ! ditto
  Commands_data:
  Data Flt Cmd¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  '
  Data   A - C ,-, Activate , Altitude, Attach, Barrier, Bingo, Cease
  Data  Chaff, Course, Cover,""
  '
  Data   D - R ,-, Deploy, Detach, Fire, Inform, Jam, Load, Mission
  Data  Proceed , Reconn, Refuel, Report,""
  '
  Data   S - Z ,-, Search, Silence , Speed, Station, Stop, Take
  Data  Turn, Use, Weapons,""
  Data ***
  '
  Menu Bar$()                 ! activate menu
  If Aclaunch!
    Print At(15,13);"¢End flight plan with STOP, BINGO, or SEARCH command.|"
  Endif
  On Menu  Gosub Flt_commands_menu_read
  Do
    On Menu
  Loop
Return   !@Flt_commands_menu
'
'                                          AIRCRFT Flt cmds Report sub-menu
Procedure Report_menu
  Menu Kill    !## kills menu; to stop use of Execute/Cancel boxes
  Restore Rreport_data
  For I=0 To 90            ! set up a loop
    Read Bar$(I)           ! read data from data field
    Exit If Bar$(I)="***"  ! until end of data field
  Next I
  Bar$(I)=""               ! tail blanks into string
  Bar$(I+1)=""             ! ditto
  Rreport_data:
  Data Report ¬C,  ??
  Data ------
  Data 1,2,3,4,5,6,""
  Data  menu ,-, <continue> , Air, ESM, On, Position, Surface, Time, Using,""
  Data ***
  Cstr$="select Report item..."
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Menu Bar$()                 ! activate menu
  On Menu  Gosub Report_menu_read
  Do
    On Menu
  Loop
```

```
Return   !@Report_menu
'
' ##############################################################################
'
'      MENU READ PROCEDURES
'
' ##############################################################################
'
'                                  Main menu selections
Procedure Main_menu_read
  Menu Off
  @Cclear_middle
  T$=F_name$   !## resets T$ to allow only one command in the string.
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))="  ??"
    @My_thesis                !alert box routine
  Endif
  If Bar$(Menu(0))=" FORCE Menu A ¬C  maneuvers   sensors   engagements"
    @Force_menu_a
  Endif
  If Bar$(Menu(0))=" FORCE Menu B ¬C  aircraft   submarines   force comms"
    @Force_menu_b
  Endif
  ' - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ASTAB orders - - - -
  '
  If Bar$(Menu(0))=" Bearing"
    T$="BEARING "
    @Bbearing
    @Show_cmd
  Endif
  '
  If Bar$(Menu(0))=" Classify"
    T$="CLASSIFY "
    @Cclassify
    @Show_cmd
  Endif
  '
  If Bar$(Menu(0))=" CPA"
    T$="CPA "
    @Ccpa
    @Show_cmd
  Endif
  '
  If Bar$(Menu(0))=" Designate "
    T$="DESIGNATE "
    @Designate_menu
  Endif
  '
  If Bar$(Menu(0))=" Drop"
    T$="DROP "
    @Ddrop
```

```
  @Show_cmd
Endif
'
If Bar$(Menu(0))=" Print"
  T$="PRINT "
  @Pprint
  @Show_cmd
Endif
'
If Bar$(Menu(0))=" Show"
  T$="SHOW "
  @Show_menu
Endif
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - - GRAPHICS orders - - - - -
If Bar$(Menu(0))=" Plot"
  @Plot_menu
Endif
If Bar$(Menu(0))=" Erase"
  @Erase_menu
Endif
If Bar$(Menu(0))=" Center"
  T$="CENTER "
  @Ccenter
  @Show_cmd
Endif
If Bar$(Menu(0))=" Radius"
  T$="RADIUS "
  @Rradius
  @Show_cmd
Endif
If Bar$(Menu(0))=" Shift"
  T$="SHIFT "
  @Sshift
  @Show_cmd
Endif
If Bar$(Menu(0))=" Label"
  T$="LABEL "
  @Llabel
  @Show_cmd
Endif
If Bar$(Menu(0))=" LOB"
  T$="LOB "
  @Llob
  @Show_cmd
Endif
If Bar$(Menu(0))=" Mark track"
  T$="MARK TRACK "
  @Mmark_track
  @Show_cmd
Endif
If Bar$(Menu(0))=" Mark bearing"
  T$="MARK BEARING "
```

```
  @Mmark_bearing
  @Show_cmd
Endif
If Bar$(Menu(0))=" Unmark track"
  T$="UNMARK TRACK "
  @Uunmark_track
  @Show_cmd
Endif
If Bar$(Menu(0))=" Place"
  T$="PLACE "
  @Pplace
  @Show_cmd
Endif
If Bar$(Menu(0))=" Cancel"
  T$="CANCEL "
  @Cancel_menu
  @Show_cmd
Endif
If Bar$(Menu(0))=" PIM"
  T$="PIM "
  @Ppim
  @Show_cmd
Endif
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - COMMS orders - - - -
If Bar$(Menu(0))=" Inform"
  T$="INFORM "
  @Show_cmd
  Cstr$="Enter text to send to player: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Print
  Form Input 75,Fi$
  T$=T$+Upper$(Fi$)
  @Show_cmd
Endif
'
If Bar$(Menu(0))=" Intell"
  T$="INTELL "
  @Iintell
  '
  @Show_cmd
Endif
If Bar$(Menu(0))=" Message "
  T$="MESSAGE "
  @Mmessage
  @Show_cmd
Endif
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - UMPIRE orders - - - -
If Bar$(Menu(0))=" Go"
  T$="GO "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Pause"
```

```
    T$="PAUSE "
    @Ppause
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" End"
    T$="END "
    @Eend
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Copy"
    T$="COPY "
    @Ccopy
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Relocate"
    T$="RELOCATE "
    @Rrelocate
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Save"
    T$="SAVE "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Time"
    T$="TIME "
    @Ttime
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Set"
    T$="SET "
    @Sset
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Enable"
    T$="ENABLE "
    @Enable_disable
    @Show_cmd
  Endif
  If Bar$(Menu(0))="  Disable"
    T$="DISABLE "
    @Enable_disable
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Expend"
    T$="EXPEND "
    @Expend_replenish
    @Show_cmd
  Endif
  If Bar$(Menu(0))="  Replenish"
    T$="REPLENISH "
    @Expend_replenish
    @Show_cmd
```

```
  Endif
Return   !@Main_menu_read
'
'                                  Force menu A selections
Procedure Force_menu_a_read
  Menu Off
  @Cclear_middle
  T$=F_name$   !## resets T$ to allow only one command in the string.
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))="  ??"
    @My_thesis                !alert box routine
  Endif
  If Bar$(Menu(0))=" MAIN Menu ¬C  astab   graphics   player comms   game"
    @Main_menu
  Endif
  If Bar$(Menu(0))=" FORCE Menu B ¬C  aircraft   submarines   force comms"
    @Force_menu_b
  Endif
  If Bar$(Menu(0))=" Select unit "
    @F_entry
    @Force_menu_a
  Endif
  ' - - - - - - - - - - - - - - - - - - - - - - - - - - - - MANEUVER orders - - - - -
  If Bar$(Menu(0))=" Course"
    T$="COURSE "
    @F_check
    @Ccourse
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Speed"
    T$="SPEED"
    @F_check
    @Sspeed
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Proceed"
    T$="PROCEED "
    @F_check
    @Pproceed
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Station"
    T$="STATION "
    @F_check
    @Sstation
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Search"
    T$="SEARCH "
    @F_check
    @Ssearch
```

```
  @Show_cmd
Endif
If Bar$(Menu(0))=" USE (plan) "
  T$="USE "
  @F_check
  @Uuse
  @Show_cmd
Endif
If Bar$(Menu(0))=" Execute (plan)"
  T$="EXECUTE "
  @F_check
  @Eexecute
  @Show_cmd
Endif
If Bar$(Menu(0))=" Enter Orders"
  Cstr$="ENTER ORDERS command not functional; need RESA input."
  Print At(12,Ytext%);Cstr$
Endif
If Bar$(Menu(0))=" Pending Orders "
  Cstr$="PENDING ORDERS command not functional; need RESA input."
  Print At(12,Ytext%);Cstr$
Endif
If Bar$(Menu(0))=" Cancel"
  T$="CANCEL "
  @F_check
  @Cancel_menu
  @Show_cmd
Endif
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - SENSORS orders - - - - -
If Bar$(Menu(0))=" Activate "
  T$="ACTIVATE "
  @F_check
  @Activate_menu
Endif
If Bar$(Menu(0))=" Silence"
  T$="SILENCE "
  @F_check
  @Silence_menu
Endif
If Bar$(Menu(0))=" Blip on"
  T$="BLIP ON "
  @F_check
  @Show_cmd
Endif
If Bar$(Menu(0))="  Blip off "
  T$="BLIP OFF "
  @F_check
  @Show_cmd
Endif
If Bar$(Menu(0))=" DECM on"
  T$="DECM ON "
  @F_check
```

```
  @Show_cmd
Endif
If Bar$(Menu(0))="  DECM off "
  T$="DECM OFF "
  @F_check
  @Show_cmd
Endif
If Bar$(Menu(0))=" RBOC on"
  T$="RBOC ON "
  @F_check
  @Show_cmd
Endif
If Bar$(Menu(0))="  RBOC off "
  T$="RBOC OFF"
  @F_check
  @Show_cmd
Endif
If Bar$(Menu(0))=" Jam"
  T$="JAM"
  @F_check
  @Jjam
  @Show_cmd
Endif
If Bar$(Menu(0))=" Cease"
  T$="CEASE"
  @F_check
  @Ccease
  @Show_cmd
Endif
If Bar$(Menu(0))=" Emcon"
  T$="EMCON"
  @F_check
  @Eemcon
  @Show_cmd
Endif
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - ENGAGE orders - - - - -
If Bar$(Menu(0))=" Weapons "
  T$="WEAPONS "
  @F_check
  @Weapons_menu
Endif
If Bar$(Menu(0))=" Fire"
  T$="FIRE "
  @F_check
  @Ffire
  @Show_cmd
Endif
If Bar$(Menu(0))=" Launch"
  T$="LAUNCH "
  @F_check
  @Llaunch
Endif
```

```
  If Bar$(Menu(0))=" Take"
    T$="TAKE "
    @F_check
    @Ttake
    @Show_cmd
  Endif
Return   !@Force_menu_A_read
'
'                                  Force menu B selections
Procedure Force_menu_b_read
  @Cclear_middle
  T$=F_name$   !## resets T$ to allow only one command in the string.
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))="  ??"
    @My_thesis                !alert box routine
  Endif
  If Bar$(Menu(0))=" MAIN Menu ¬C  astab   graphics   player comms   game"
    @Main_menu
  Endif
  If Bar$(Menu(0))=" FORCE Menu A ¬C  maneuvers   sensors   engagements"
    @Force_menu_a
  Endif
  If Bar$(Menu(0))=" Select unit "
    @F_entry
    @Force_menu_b
  Endif
  ' - - - - - - - - - - - - - - - - - - - - - AIRCRFT selections - - - - - - -
  If Bar$(Menu(0))=" Launch "
    T$="LAUNCH "
    @F_check
    @Ac_launch
  Endif
  '
  If Bar$(Menu(0))=" Flight Cmds "
    @F_check
    Cstr2$=""
    @Flt_commands_menu
  Endif
  '
  If Bar$(Menu(0))=" Alert"
    T$="ALERT"
    @F_check
    @Aalert
  Endif
  '
  If Bar$(Menu(0))=" Close"
    T$="CLOSE"
    @F_check
    @Show_cmd
    Inc Ytext%
    @Tc_choice
```

```
Endif
'
If Bar$(Menu(0))=" Handover"
  T$="HANDOVER"
  @F_check
  @Hhandover
Endif
'
If Bar$(Menu(0))=" Open"
  T$="OPEN"
  @F_check
  @Show_cmd
  Inc Ytext%
  @Tc_choice
Endif
'
If Bar$(Menu(0))=" Orbit"
  T$="ORBIT"
  @F_check
  @Oorbit
Endif
'
If Bar$(Menu(0))=" Recall"
  T$="RECALL"
  @F_check
  @Show_cmd
  Cstr$="All flights."
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Tc_choice
Endif
'
If Bar$(Menu(0))=" Recover"
  T$="RECOVER"
  @F_check
  @Show_cmd
  Cstr$="All flights."
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Tc_choice
Endif
' - - - - - - - - - - - - - - - - - - - - SUBMRINE selections - - - - - - -
If Bar$(Menu(0))=" Depth"
  T$="DEPTH "
  @F_check
  Cstr$="Enter depth (60-9999 ft): "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=60
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+Pnum$
```

```
    Ytext%=Crslin
    Inc Ytext%
    @Tc_choice
  Endif
  '
  If Bar$(Menu(0))=" Surface"
    T$="SURFACE"
    @F_check
    @Tc_choice
  Endif
  '
  If Bar$(Menu(0))=" Periscope "
    T$="PERISCOPE"
    @F_check
    Cstr$="Come to periscope depth."
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Inc Ytext%
    @Tc_choice
  Endif
  '
  If Bar$(Menu(0))=" Fire"
    T$="FIRE "
    @F_check
    @Ffire
  Endif
  '
  If Bar$(Menu(0))=" Mode"
    T$="MODE"
    @F_check
    Alert 2,"   Which mode? ",0,"Diesel Electric",A
    If A=1 Then
      T$=T$+" DIESEL"
    Endif
    If A=2 Then
      T$=T$+" ELECTRIC"
    Endif
    Clr A
    Inc Ytext%
    @Tc_choice
  Endif
  '
  If Bar$(Menu(0))=" Mast"
    T$="MAST"
    @F_check
    Alert 2,"  Select... ",0,"  Down   Up",A
    If A=1 Then
      T$=T$+" DOWN"
    Endif
    If A=2 Then
      T$=T$+" UP"
    Endif
    Clr A
```

```
      Inc Ytext%
      @Tc_choice
    Endif
    '
    If Bar$(Menu(0))=" Deploy"
      T$="DEPLOY "
      @F_check
      @Ddeploy
    Endif
    '
    If Bar$(Menu(0))=" Retrieve"
      T$="RETRIEVE "
      @F_check
      @Ddeploy  !## used also for "Retrieve"
    Endif
    ' - - - - - - - - - - - - - - - - - - - - - COMMTEXT selections - - - - - - -
    If Bar$(Menu(0))=" Commtext "
      T$="COMMTEXT "
      @F_check
      @Ccommtext
    Endif
    '
    If Bar$(Menu(0))=" Embark"
      T$="EMBARK "
      @F_check
      @Eembark
    Endif
    '
    If Bar$(Menu(0))=" Report"
      T$="REPORT "
      @F_check
      @Show_cmd
      @Report_menu
    Endif
    '
    If Bar$(Menu(0))=" Circuit"
      T$="CIRCUIT "
      @F_check
      @Show_cmd
      Cstr$="Enter circuit number: "
      Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
      Form Input 5,Fi$
      T$=T$+Upper$(Fi$)
    Endif
    '
    @Show_cmd
    @Force_menu_b
Return   !@Force_menu_b_read
'
'                                   Designate menu selections
Procedure Designate_menu_read
  Menu Off
```

```
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" Enemy"
    T$=T$+"ENEMY "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If Bar$(Menu(0))=" Friendly "
    T$=T$+"FRIENDLY "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If Bar$(Menu(0))=" Neutral"
    T$=T$+"NEUTRAL "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If Bar$(Menu(0))=" Unknown"
    T$=T$+"UNKNOWN "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  @Show_cmd
  @Main_menu
Return   !@Designate_menu_read
'
'                                        'Show' menu selections
Procedure Show_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" AAWC"
    T$=T$+"AAWC "
    @Af_choice    !## AIR/FLIGHT choice
    @Display_menu
  Endif
  '
  If Bar$(Menu(0))=" Active"
```

```
  T$=T$+"ACTIVE "
  @Show_cmd
  Alert 2,"    Select...  ",0,"Sonar  Tracks ",A
  If A=1
    T$=T$+"SONAR "
  Endif
  If A=2
    T$=T$+"TRACKS "
  Endif
  Clr A
  @Show_cmd
  @Display_menu    !## Sub-menu for Show menu
Endif
'
If Bar$(Menu(0))=" Air"
  T$=T$+"AIR "
  @Show_cmd
  @Show_air_menu
Endif
'
If Bar$(Menu(0))=" ASUWC"
  T$=T$+"ASUWC "
  @Af_choice    !## AIR/FLIGHT choice
Endif
'
If Bar$(Menu(0))=" ASWC"
  T$=T$+"ASWC "
  @Af_choice    !## AIR/FLIGHT choice
Endif
'
If Bar$(Menu(0))=" Bogey (tote & cap)"
  T$=T$+"BOGEY "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Continuation (of next page) "
  T$=T$+"CONTINUATION "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Damage (& reconn info) "
  T$=T$+"DAMAGE "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" ESM"
  T$=T$+"ESM "
  Alert 2,"       ESM...       ",0,"Air Surface Tracks",A
  If A=1
```

```
    T$=T$+"AIR "
  Endif
  If A=2
    T$=T$+"SURFACE "
  Endif
  If A=3
    T$=T$+"TRACKS "
  Endif
  Clr A
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" EWC"
  T$=T$+"EWC "
  @Af_choice    !## AIR/FLIGHT choice
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Flight"
  T$=T$+"FLIGHT "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Force"
  T$=T$+"FORCE "
  @Show_cmd
  Cstr$="Enter force name: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)+" "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" HFDF"
  T$=T$+"HFDF "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Intell (spot reports)"
  T$=T$+"INTELL "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Passive (sonar tracks)"
  T$=T$+"PASSIVE "
  @Show_cmd
  @Display_menu
```

```
Endif
'
If Bar$(Menu(0))=" Reporting (policies)"
  T$=T$+"REPORTING "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Ship"
  T$=T$+"SHIP "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Shore"
  T$=T$+"SHORE "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" SOSUS (tracks)"
  T$=T$+"SOSUS "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Submarine"
  T$=T$+"SUBMARINE "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Surface (tracks)"
  T$=T$+"SURFACE "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Surveillance (satellites) "
  T$=T$+"SURVEILLANCE "
  @Show_cmd
  @Display_menu
Endif
'
If Bar$(Menu(0))=" Weather"
  T$=T$+"WEATHER "
  @Show_cmd
  @Display_menu
Endif
'
@Show_cmd
@Main_menu
```

```
Return   !@Show_menu_read
'
'                                        'Display' menu selections
Procedure Display_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  Cstr$="Enter view number: "
  '
  If Bar$(Menu(0))=" Blue"
    T$=T$+"BLUE "
    @Va_entry
  Endif
  '
  If Bar$(Menu(0))=" Orange"
    T$=T$+"ORANGE "
    @Va_entry
  Endif
  '
  If Bar$(Menu(0))=" Neutral"
    T$=T$+"NEUTRAL "
  Endif
  '
  If Bar$(Menu(0))=" <astab>"
    Cstr$="Enter ASTAB number: "
    @Va_entry
  Endif
  '
  @Show_cmd
  @Main_menu
Return   !@Display_menu_read
'
'                                        'Display' menu selections
Procedure Show_air_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  Cstr$="Enter ASTAB number: "
  '
  If Bar$(Menu(0))=" Alert"
    T$=T$+"ALERT "
    @Va_entry
  Endif
  '
  If Bar$(Menu(0))=" Availability "
    T$=T$+"AVAILABILITY "
    @Va_entry
  Endif
  '
  If Bar$(Menu(0))=" Events"
```

```
    T$=T$+"EVENTS "
    @Va_entry
  Endif
  '
  If Bar$(Menu(0))=" Tracks"
    T$=T$+"TRACKS "
    @Va_entry
  Endif
  '
  @Show_cmd
  @Main_menu
Return   !@Show_air_menu_read
'
' - - - - - - - - - - Common entry for AIR/FLIGHT choices from Show menu - - -
Procedure Af_choice
  @Show_cmd
  Alert 2,"    Select...  ",0,"Air  Flight ",A
  If A=1
    T$=T$+"AIR "
  Endif
  If A=2
    T$=T$+"FLIGHT "
  Endif
  Clr A
Return
'
'  - - - - - - - Common entry for View/ASTAB number from Display menu read - -
Procedure Va_entry
  @Show_cmd
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=0
  Hilim%=6
  Numlen%=1
  @Number_entry
  T$=T$+Pnum$
Return
'
' - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
'                                  Plot_Erase menu selections
Procedure Plot_erase_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" MAIN Menu ¬C  astab   graphics   player comms   game"
    @Main_menu
  Endif
  If Bar$(Menu(0))=" FORCE Menu A ¬C  maneuvers   sensors   engagements"
    @Force_menu_a
  Endif
  If Bar$(Menu(0))=" FORCE Menu B ¬C  aircraft   submarines   force comms"
    @Force_menu_b
```

```
Endif
If Bar$(Menu(0))=" All"
  T$=T$+"ALL "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Blue"
  T$=T$+"BLUE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Orange"
  T$=T$+"ORANGE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Own"
  T$=T$+"OWN "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Boundaries "
  T$=T$+"BOUNDARIES "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Chaff"
  T$=T$+"CHAFF "
  @Show_cmd
Endif
If Bar$(Menu(0))=" LOB"
  T$=T$+"LOB "
  @Show_cmd
Endif
If Bar$(Menu(0))=" PIM"
  T$=T$+"PIM "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Regions"
  T$=T$+"REGIONS "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Rivers"
  T$=T$+"RIVERS "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Sonobuoy"
  T$=T$+"SONOBUOY "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Speed"
  T$=T$+"SPEED "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Survsat"
  T$=T$+"SURVSAT "
  @Show_cmd
```

```
  Endif
  If Bar$(Menu(0))=" Track"
    T$=T$+"TRACK "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Station"
    T$=T$+"STATION "
    @Show_cmd
  Endif
  '
  @Main_menu    !returns to main menu, allowing only one plot/erase choice
Return   !@Plot_erase_menu_read
'
'                                    Cancel menu selections
Procedure Cancel_menu_read
  Menu Off
  @F_check
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" MAIN Menu ¬C  astab   graphics   player comms   game"
    @Main_menu
  Endif
  If Bar$(Menu(0))=" FORCE Menu A ¬C  maneuvers   sensors   engagements"
    @Force_menu_a
  Endif
  If Bar$(Menu(0))=" FORCE Menu B ¬C  aircraft   submarines   force comms"
    @Force_menu_b
  Endif
  If Bar$(Menu(0))=" Activate "
    T$=T$+"ACTIVATE "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" All"
    T$=T$+"ALL "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Altitude"
    T$=T$+"ALTITUDE "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Attach"
    T$=T$+"ATTACH "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Barrier"
    T$=T$+"BARRIER "
    @Show_cmd
  Endif
  If Bar$(Menu(0))=" Bingo"
    T$=T$+"BINGO "
    @Show_cmd
```

```
Endif
If Bar$(Menu(0))=" Blip"
  T$=T$+"BLIP "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Cease"
  T$=T$+"CEASE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Chaff"
  T$=T$+"CHAFF "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Circle "
  T$=T$+"CIRCLE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Course"
  T$=T$+"COURSE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Cover"
  T$=T$+"COVER "
  @Show_cmd
Endif
If Bar$(Menu(0))=" DECM"
  T$=T$+"DECM "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Deploy"
  T$=T$+"DEPLOY"
  @Show_cmd
Endif
If Bar$(Menu(0))=" Depth"
  T$=T$+"DEPTH "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Detach"
  T$=T$+"DETACH "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Emcon"
  T$=T$+"EMCON "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Execute "
  T$=T$+"EXECUTE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Fire"
  T$=T$+"FIRE "
  @Show_cmd
```

```
Endif
If Bar$(Menu(0))=" Grid"
  T$=T$+"GRID "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Jam"
  T$=T$+"JAM "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Launch"
  T$=T$+"LAUNCH "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Mast"
  T$=T$+"MAST "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Mission"
  T$=T$+"MISSION "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Mode"
  T$=T$+"MODE "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Orbit"
  T$=T$+"ORBIT "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Proceed"
  T$=T$+"PROCEED "
  @Show_cmd
Endif
If Bar$(Menu(0))=" RBOC"
  T$=T$+"RBOC "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Recall"
  T$=T$+"RECALL "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Reconn"
  T$=T$+"RECONN "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Recover"
  T$=T$+"RECOVER "
  @Show_cmd
Endif
If Bar$(Menu(0))=" Refuel"
  T$=T$+"REFUEL "
  @Show_cmd
```

```
    Endif
    If Bar$(Menu(0))=" Retrieve "
      T$=T$+"RETRIEVE "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Search"
      T$=T$+"SEARCH "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Silence "
      T$=T$+"SILENCE "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Speed"
      T$=T$+"SPEED "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Station"
      T$=T$+"STATION "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Take"
      T$=T$+"TAKE "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Turn"
      T$=T$+"TURN "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Weapons"
      T$=T$+"WEAPONS "
      @Show_cmd
    Endif
    If Bar$(Menu(0))=" Xmark"
      T$=T$+"XMARK "
      @Show_cmd
    Endif
    '
    @Main_menu       !returns to main menu, allowing only one "Cancel" choice
Return   !@Cancel_menu_read
'
'                                                  WEAPONS sub-menu selections
Procedure Weapons_menu_read
    '                          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    '                          !    Weapons menu(0) codes
    '                          !              FREE  /  TIGHT
    '                          ! air           12       22
    '                          ! surface       13       23
    '                          ! submarine     14       24
    '                          ! all           15       25
    '                          ! enemy         16       26
    '                          ! nuclear       17       27
```

```
'                          ! conventional 18       28
'                          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Menu Off
@Cclear_middle
'
If Firsttime!          !## Keeps from repeating "Weapons Tight/Free"
  If Menu(0)>10 And Menu(0)<20
    T$=T$+"FREE "
    N%=20     !## menu() numbers
  Endif
  If Menu(0)>20 And Menu(0)<30
    T$=T$+"TIGHT "
    N%=10
  Endif
  For I%=N%+2 To N%+8  !## disables either Tight or Free menu items
    Menu I%,2          !
  Next I%              !
  Firsttime!=False    !## keeps from re-entering this if-loop
Endif
'
If Bar$(Menu(0))=" Nuclear"
  T$=T$+"NUCLEAR "
  Menu 17,2
  Menu 18,2
  Menu 27,2
  Menu 28,2
  @Partial     !## allows use of same menu to complete Weapons command
Endif
'
If Bar$(Menu(0))=" Conventional "
  T$=T$+"CONVENTIONAL "
  Menu 17,2
  Menu 18,2
  Menu 27,2
  Menu 28,2
  @Partial
Endif
'
If Bar$(Menu(0))=" Enemy"
  T$=T$+"ENEMY "
  Menu 16,2
  Menu 17,2
  Menu 18,2
  Menu 26,2
  Menu 27,2
  Menu 28,2
  @Partial     !## allows completing string using same menu
Endif
'
If Bar$(Menu(0))=" Surface"
  T$=T$+"SURFACE"
  @Tc_choice
```

```
  Endif
  If Bar$(Menu(0))=" Submarine"
    T$=T$+"SUBMARINE"
    @Tc_choice
  Endif
  If Bar$(Menu(0))=" Air"
    T$=T$+"AIR"
    @Tc_choice
  Endif
  If Bar$(Menu(0))=" All"
    T$=T$+"ALL"
    @Tc_choice
  Endif
  '
  If Aclaunch!          !## Launch seq is being used
    @Show_cmd
    @Flt_commands_menu
  Else
    @Show_cmd
    @Force_menu_a  !## will branch to Force_menu_a if Aclaunch! False.
  Endif
Return   !@Weapons_menu_read
'
'                                          Launch CRUISE sub-menu selections
Procedure Cruise_menu_read
  Menu Off
  @Cclear_middle
  Menu Kill   !## allows menu to only be used once.
  Goto Lc1    !## returns to where it was before branching to Cruise_menu.
Return   !@Weapons_menu_read
'
'                                       Activate menu selections
Procedure Activate_menu_read
  Menu Off
  @Cclear_middle
  '
  If Bar$(Menu(0))=" Air"
    T$=T$+"AIR"
    Cstr$=" search radar"
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Inc Ytext%
    @Use_time_choice
  Endif
  '
  If Bar$(Menu(0))=" Approach "
    T$=T$+"APPROACH"
    @Show_cmd
    Cstr$=" radar "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Inc Ytext%
    @Use_time_choice
  Endif
```

```
  '
  If Bar$(Menu(0))=" Emitter"
    T$=T$+"EMITTER"
    @Eemitter
  Endif
  '
  If Bar$(Menu(0))=" ESM"
    T$=T$+"ESM"
    @Use_time_choice
  Endif
  '
  If Bar$(Menu(0))=" Radar"
    T$=T$+"RADAR"
    @Show_cmd
    Cstr$=" (air/surface/approach) "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Inc Ytext%
    @Use_time_choice
  Endif
  '
  If Bar$(Menu(0))=" Sonar"
    T$=T$+"SONAR "
    @Sonar_menu
  Endif
  '
  If Bar$(Menu(0))=" Surface"
    T$=T$+"SURFACE"
    @Show_cmd
    Cstr$=" search radar "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Inc Ytext%
    @Use_time_choice
  Endif
  '
  If Bar$(Menu(0))=" Survsat"
    T$=T$+"SURVSAT "
    @Ssurvsat
  Endif
  '
  If Aclaunch!          !## Launch seq is being used
    @Show_cmd
    @Flt_commands_menu
  Else
    @Show_cmd
    @Force_menu_a  !## will branch to Force_menu_a if Aclaunch! False.
  Endif
Return  !@Activate_menu_read
'
'                                  Silence menu selections
Procedure Silence_menu_read
  Menu Off
  @Cclear_middle
```

```
'
If Bar$(Menu(0))=" Air"
  T$=T$+"AIR"
  Cstr$=" search radar"
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" Approach "
  T$=T$+"APPROACH"
  Cstr$=" radar "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" Emitter"
  T$=T$+"EMITTER"
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" ESM"
  T$=T$+"ESM"
  Cstr$=" equipment "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" Radar"
  T$=T$+"RADAR"
  Cstr$=" (air/surface equipment) "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" Sonar"
  T$=T$+"SONAR"
  Cstr$=" equipment "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
Endif
'
If Bar$(Menu(0))=" Surface"
  T$=T$+"SURFACE"
  Cstr$=" search radar equipment "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Inc Ytext%
  @Name_time_choice
```

```
  Endif
  '
  If Bar$(Menu(0))=" Survsat"
    T$=T$+"SURVSAT "
    @Show_cmd
    Cstr$="Enter satellite name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)+" "
    '
    @Show_cmd
    @Cclear_middle
    Cstr$="Enter force name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
    @Tc_choice
  Endif
  '
  If Aclaunch!          !## Launch seq is being used
    @Show_cmd
    @Flt_commands_menu
  Else
    @Show_cmd
    @Force_menu_a  !## will branch to Force_menu_a if Aclaunch! False.
  Endif
Return  !@Silence_menu_read
'
'                                              Sonar menu selections
Procedure Sonar_menu_read
  Menu Off
  @Cclear_middle
  '
  If Bar$(Menu(0))=" BB"
    T$=T$+"BB"
  Endif
  '
  If Bar$(Menu(0))=" CZ"
    T$=T$+"CZ"
  Endif
  '
  If Bar$(Menu(0))=" DP"
    T$=T$+"DP"
  Endif
  '
  @Use_time_choice
  '
  If Aclaunch!          !## Launch seq is being used
    @Show_cmd
    @Flt_commands_menu
  Else
    @Show_cmd
```

```
    @Force_menu_a  !## will branch to Force_menu_a if Aclaunch! False.
  Endif
Return
'
'                                          Mission menu selections
Procedure Mission_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" none"
    T$=T$+"MISSION NONE "
    @Show_cmd
  Endif
  '
  If Bar$(Menu(0))=" AEW"
    Cstr2$="MISSION AEW"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Airtanker"
    Cstr2$="MISSION AIRTANKER"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" ASW"
    Cstr2$="MISSION ASW"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" CAP"
    Cstr2$="MISSION CAP"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Decoy"
    Cstr2$="MISSION DECOY"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" EW"
    Cstr2$="MISSION EW"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Jammer"
    Cstr2$="MISSION JAMMER"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Reconn"
    Cstr2$="MISSION RECONN"
```

```
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Relay"
    Cstr2$="MISSION RELAY"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Rescue"
    Cstr2$="MISSION RESCUE"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Search"
    Cstr2$="MISSION SEARCH"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Strcap"
    Cstr2$="MISSION STRCAP"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Strike"
    Cstr2$="MISSION STRIKE"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Sttanker"
    Cstr2$="MISSION STTANKER"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Surcap"
    Cstr2$="MISSION SURCAP"
    @Mission_common
  Endif
  '
  If Bar$(Menu(0))=" Surveillance "
    Cstr2$="MISSION SURVEILLANCE"
    @Mission_common
  Endif
  '  - - - - - - If Launch seq is being used, Aclaunch! is True. - - - - -
  If Aclaunch!
    @Flt_commands_menu
  Else
    @Force_menu_b  !## will branch to Force_menu_b if Aclaunch! False.
  Endif
Return  !@Mission_menu_read
'
'                                        A/C Commands menu selections
Procedure Flt_commands_menu_read
```

```
Menu Off
@Cclear_middle
'
If Aclaunch!          !## Launch seq is being used
  T$=T$+" "
Endif
' - - - - - - - - - - - - - menu items with existing Procedures - - - - - -
'
If Bar$(Menu(0))=" Activate "
  @Activate_menu
Endif
'
If Bar$(Menu(0))=" Altitude"
  T$=T$+"ALTITUDE "
  @Altitude_entry
Endif
'
If Bar$(Menu(0))=" Cease"
  T$=T$+"CEASE"
  @Ccease
Endif
'
If Bar$(Menu(0))=" Course"
  T$=T$+"COURSE "
  @Ccourse
Endif
'
If Bar$(Menu(0))=" Fire"
  T$=T$+"FIRE "
  @Ffire
Endif
'
If Bar$(Menu(0))=" Jam"
  T$=T$+"JAM "
  @Jjam
Endif
'
If Bar$(Menu(0))=" Proceed "
  T$=T$+"PROCEED "
  @Pproceed
Endif
'
If Bar$(Menu(0))=" Silence "
  @Silence_menu
Endif
'
If Bar$(Menu(0))=" Speed"
  T$=T$+"SPEED"
  @Sspeed
Endif
'
If Bar$(Menu(0))=" Station"
```

```
  T$=T$+"STATION "
  @Sstation
Endif
'
If Bar$(Menu(0))=" Take"
  T$=T$+"TAKE "
  @Ttake
Endif
'
If Bar$(Menu(0))=" Use"
  T$=T$+"USE "
  @Uuse
Endif
'
If Bar$(Menu(0))=" Weapons"
  If Aclaunch!
    T$=T$+"WEAPONS "
  Else
    T$="WEAPONS "
    @F_check
  Endif
  Firsttime!=True
  @Weapons_menu
Endif
'
' - - - - - - - - - - - - - menu items with new Procedures - - - - - - - - -
'
If Bar$(Menu(0))=" Attach"
  T$=T$+"ATTACH"
  @Aattach
Endif
'
If Bar$(Menu(0))=" Barrier"
  T$=T$+"BARRIER"
  @Bbarrier
Endif
'
If Bar$(Menu(0))=" Chaff"
  T$=T$+"CHAFF "
  @Cchaff
Endif
'
If Bar$(Menu(0))=" Cover"
  T$=T$+"COVER "
  @Ccover
Endif
'
If Bar$(Menu(0))=" Deploy"
  T$=T$+"DEPLOY "
  @Ddeploy
Endif
'
```

```
If Bar$(Menu(0))=" Detach"
  T$=T$+"DETACH"
  @Show_cmd
  Cstr$="From collective flight."
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  @Time_entry
Endif
'
If Bar$(Menu(0))=" Inform"
  T$=T$+"INFORM "
  @Show_cmd
  Cstr$="Enter text to send to player. "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 20,Fi$
  T$=T$+Upper$(Fi$)
Endif
'
If Bar$(Menu(0))=" Load"
  T$=T$+"LOAD"
  @Show_cmd
  @Ac_load
Endif
'
If Bar$(Menu(0))=" Mission"
  @Mission_menu
Endif
'
If Bar$(Menu(0))=" Reconn"
  T$=T$+"RECONN "
  @Rreconn
Endif
'
If Bar$(Menu(0))=" Refuel"
  T$=T$+"REFUEL "
  @Rrefuel
Endif
'
If Bar$(Menu(0))=" Report"
  T$=T$+"REPORT "
  @Show_cmd
  @Report_menu
Endif
'
If Bar$(Menu(0))=" Turn"
  T$=T$+"TURN "
  @Show_cmd
  @Course_entry
  @Show_cmd
  @Time_entry
Endif
'
' - - - - - - - - - - - - menu items that end the Launch sequence - - - - -
```

```
  '
  If Bar$(Menu(0))=" Stop"
    Aclaunch!=False   !## resets Launch sequence flag
    T$=T$+"STOP "
  Endif
  '
  If Bar$(Menu(0))=" Bingo"
    Aclaunch!=False   !## resets Launch sequence flag
    T$=T$+"BINGO "
  Endif
  '
  If Bar$(Menu(0))=" Search"
    Aclaunch!=False   !## resets Launch sequence flag
    T$=T$+"SEARCH "
    @Ssearch
  Endif
  '
  If Aclaunch!          !## Launch seq is being used
    @Show_cmd
    @Flt_commands_menu
  Else
    @Show_cmd
    @Force_menu_b  !## will branch to Force_menu_b if Aclaunch! False.
  Endif
Return  !@Flt_commands_menu_read
'
'
'                                Flt Commands REPORT menu selections
Procedure Report_menu_read
  Menu Off
  @Cclear_middle
  @Cclear_command_box   !## clears it even if cmd not Cancelled or Executed
  '
  If Bar$(Menu(0))=" Air"
    T$=T$+"AIR "
    @Aair
  Endif
  '
  If Bar$(Menu(0))=" ESM"
    T$=T$+"ESM "
    @Eesm
  Endif
  '
  If Bar$(Menu(0))=" On"
    T$=T$+"ON "
    @Oon
  Endif
  '
  If Bar$(Menu(0))=" Position"
    T$=T$+"POSITION "
    @Pposition
  Endif
```

```
  '
  If Bar$(Menu(0))=" Surface"
    T$=T$+"SURFACE "
    @Ssurface
  Endif
  '
  If Bar$(Menu(0))=" Time"
    @Tttime
  Endif
  '
  If Bar$(Menu(0))=" Using"
    T$=T$+"USING "
    @Uusing
  Endif
  '
  @Show_cmd
  @Force_menu_b
Return  !@Report_menu_read
'
' ###########################################################################
'
'      GENERAL WORKING PROCEDURES
'
' ###########################################################################
'
' ------------------- Proc to test keyboard scan & window use -----------
Procedure Help_key_test
  If Menu(14)=25088     !## 25088 is the scan code for Help key
    W1x=300    ! maximum 638 (639 will cause the window to NOT reset)
    W1y=189    ! maximum 189 (menu bar takes 10 pix lines of screen)
    Get 0,0,W1x+1,W1y+1,Savewindow1$
    Openw 1,W1x,W1y
    Titlew 1,"Message about Help."
    Clearw 1
    Print "This is a help paragraph.  It can be set up so that it is right"
    Print "where you want it.  I am testing the Help Key."
    Print "  Free RAM = ";Fre(0)
    Print "  Press a key to continue."
    Repeat
    Until Inkey$<>""
    Closew 1
    Put 0,0,Savewindow1$
  Endif
  Closew 0    !## Resets screen after other windows are used.
Return
'
'                    DIOX (Dialog box ) common procedure
'         Used as a "shell" procedure to help create dialog boxes
'
Procedure
Draw_text_in_box(Ch$,X_text,Y_text,Style,Char_color,Char_size,Hborder,Vborder,-
Thick,Inverse)
```

```
  Local
Offset,Width,Height,Fatness,Round,Seethru,Xhot_upper,Yhot_upper,Xhot_lower,-
Yhot_lower,Temp$
  If X_text<0
    Round=True
  Else
    Round=False
  Endif
  If Y_text<0
    Seethru=True
  Else
    Seethru=False
  Endif
  X_text=Abs(X_text)
  Y_text=Abs(Y_text)
  If Xbios(4)=1 And Char_size=1
    Char_size=6
  Else
    If Xbios(4)=2 And Char_size=1
      Char_size=13
    Endif
  Endif
  If Char_size=32
    Height=32
    Width=16
    Offset=4
  Else
    If Char_size=13
      Height=16
      Width=8
      Offset=3
    Else
      If Char_size=6
        Height=8
        Width=8
        Offset=1
      Else
        If Char_size=4
          Height=7
          Width=6
          Offset=2
        Endif
      Endif
    Endif
  Endif
'
If Xbios(4)=2
  Strip=(0)
Else
  Strip=(-8)
Endif
Xhot_upper=X_text-Hborder-Thick+1
```

```
  Yhot_upper=Y_text+Offset-Height-Vborder-Thick+20+Strip
  Xhot_lower=X_text+Len(Ch$)*Width+Hborder+Thick-1
  Yhot_lower=Y_text+Offset+Vborder+Thick+18+Strip
  '
  Deftext Char_color,Style,0,Char_size
  Graphmode 1
  '
  If Thick>0
    If Inverse
      Deffill Char_color,1,
    Else
      Deffill 0,0,0
    Endif
    Color 1
    '
    If Round
      If Seethru
        Rbox
(X_text-Hborder)-1,(Y_text+Offset-Height-Vborder)-1,(X_text+Len(Ch$)*Width-
+Hborder)+1,(Y_text+Offset+Vborder)+1
      Else
        Prbox
(X_text-Hborder)-1,(Y_text+Offset-Height-Vborder)-1,(X_text+Len(Ch$)*Width-
+Hborder)+1,(Y_text+Offset+Vborder)+1
      Endif
    Else
      If Seethru
        Box
(X_text-Hborder)-1,(Y_text+Offset-Height-Vborder)-1,(X_text+Len(Ch$)*Width-
+Hborder)+1,(Y_text+Offset+Vborder)+1
      Else
        Pbox
(X_text-Hborder)-1,(Y_text+Offset-Height-Vborder)-1,(X_text+Len(Ch$)*Width-
+Hborder)+1,(Y_text+Offset+Vborder)+1
      Endif
    Endif
    '
    For Fatness=1 To Thick
      '
      If Round
        Rbox
(X_text-Hborder)-Fatness,(Y_text+Offset-Height-Vborder)-Fatness,(X_text+Len(Ch$)
*Width+Hborder)+Fatness,(Y_text+Offset+Vborder)+Fatness
      Else
        Box
(X_text-Hborder)-Fatness,(Y_text+Offset-Height-Vborder)-Fatness,(X_text+Len(Ch$)
*Width+Hborder)+Fatness,(Y_text+Offset+Vborder)+Fatness
      Endif
    Next Fatness
    '
  Endif
```

```
  '
  If Seethru
    Graphmode 2
    Text X_text,Y_text,Ch$
    If Inverse
      Get Xhot_upper,Yhot_upper,Xhot_lower,Yhot_lower,Temp$
      Put Xhot_upper,Yhot_upper,Temp$,12
    Endif
    Goto Buttonend1
  Endif
  '
  If Inverse And Thick>0
    Graphmode 3
  Else
    If Inverse And Thick<=0
      Graphmode 4
    Else
      If Not (Inverse)
        Graphmode 2
      Endif
    Endif
  Endif
  Text X_text,Y_text,Ch$
  Buttonend1:
  Graphmode 1
Return   ! DIOX common procedure
'
'         save selected 'blank' areas of screen to use as "erasers" later
'
Procedure Save_blanks
  Cls
  Get 0,11,639,140,Middle_box$
  Get 9,162,625,194,Command_box$
Return
'
'                         response to ?? dialog box
Procedure My_thesis
  Menu Off
  Mtxt$=" Thesis by LCDR G.L.Yungk  advisor - CDR J. Stewart"
  Alert 1,Mtxt$,1," OK ",A
  Clr A
Return
'
'                         Initialization/dimensioning
Procedure Init
  Setcolor 2,1911     !## Turns green background white. (for windowing)
  Dim Bar$(90)        !## Main menu bar.
  F_name$=""          !## Common start to "Force" commands
  Ytext%=8            !## Common line to start text on.
  Aclaunch!=False     !## allows use of individual Procs/menus in Launch seq.
  Lat_str$="Enter latitude   (0-89N or S): "  !## To allow use of err-chkng
  Long_str$="Enter longitude (0-180E or W): " !##  with common Procedures.
```

```
Return
'
'          Store words/phrases in a string & prints in Command box
'
Procedure Show_cmd
  @Cclear_command_box
  If Len(T$)<=100                !## 100 characters, 2 lines max
    Deftext 1,0,0,9              !## text size allows 50 characters per line
    Text 15,175,Mid$(T$,0,50)    !## first line in command box
    Text 15,190,Mid$(T$,51)      !## second line in command box
  Else                           !## 101 - 225 characters, 3 lines max
    Deftext 1,0,0,6              !## normal text, allows 75 characters per line
    Text 15,170,Mid$(T$,0,75)
    Text 15,180,Mid$(T$,76,75)
    Text 15,190,Mid$(T$,151,75)
  Endif
  Deftext 1,0,0,6              !## resets text to normal size & color
Return
'
'                     Draw the Command, EXECUTE, & CANCEL boxes
Procedure Draw_box
  Deftext 1,0,0,8
  Text 460,155,"EXECUTE command"
  Text 15,155,"CANCEL command"
  Color 2
  Defline 1,2,0,0           ! line for EXECUTE box
  Box 450,142,630,160       ! EXECUTE box
  Box 5,142,175,160         ! CANCEL box
  Defline 1,5,0,0           ! line for Command box
  Box 5,160,630,196         ! Command box
  Color 1                   ! resets color to black
  Defline 1,1,0,0           ! resets line to normal width
  Deftext 1,0,0,6           ! resets text to normal size & color
Return   !@Draw_box
'
'                         Test for mouse In the EXECUTE box
Procedure Inbox_execute
  Deftext 2,0,0,8
  Text 460,155,"EXECUTE"
  Deftext 1,0,0,6          !## resets text type to normal
  Sound 1,15,5,4,3
  Sound 1,15,8,4,3
  Sound 1,15,5,4,3
  Sound 1,0,0,0
  On Menu Obox 1,450,142,180,18 Gosub Outbox_execute
  On Menu Button 1,1,1 Gosub Send_string
  Do
    On Menu
  Loop
Return
'
'                         Test for mouse Out of EXECUTE box
```

```
Procedure Outbox_execute
  Deftext 1,0,0,8
  Text 460,155,"EXECUTE"
  Deftext 1,0,0,6          !## resets text type to normal
  On Menu Ibox 1,450,142,180,18 Gosub Inbox_execute
Return
'
'                            Test for mouse In CANCEL box
Procedure Inbox_cancel
  Sound 1,15,10,4
  Deftext 2,0,0,8
  Text 15,155,"CANCEL"
  Deftext 1,0,0,6          !## resets text type to normal
  Sound 1,0,0,0
  On Menu Obox 2,5,142,170,18 Gosub Outbox_cancel
  On Menu Button 1,1,1 Gosub Cancel_string
  Do
    On Menu
  Loop
Return
'
'                          Test for mouse Out of CANCEL box
Procedure Outbox_cancel
  Deftext 1,0,0,8
  Text 15,155,"CANCEL"
  Deftext 1,0,0,6          !## resets text type to normal
  On Menu Ibox 2,5,142,170,18 Gosub Inbox_cancel
Return
'
'                                Clear command box
Procedure Cclear_command_box
  Put 9,162,Command_box$,3
Return
'
'                            Send completed "order" to RESA
Procedure Send_string
  If T$<>""    !## string must have something in it.
    @Cclear_middle
    @Cclear_command_box
    '
    '  Open "",#1,"AUX:"
    '
    If Out?(1)=-1        !##  RS232 port is (1).  -1 is ready; 0 is not.
      Print At(1,4);"sending: ";
      For I=1 To Len(T$)           !## maybe needs Len(T$)-1
        Pause 1
        Out 1,Asc(Mid$(T$,I,1))  !## output to RS232 port
        Out 2,Asc(Mid$(T$,I,1))  !## output to screen
      Next I
    Endif
    '
    '  Close #1
```

```
    '
    T$=""
  Endif
Return
'
'                          Cancel order/command & clear string
Procedure Cancel_string
  @Cclear_middle
  @Cclear_command_box
  Print At(25,10);"command cancelled."
  T$=""
Return
'
'                              clear middle of screen
Procedure Cclear_middle
  Put 0,11,Middle_box$,3
  Ytext%=8                 !## To reset first text line to row 8.
Return
'
' --------------------------- Error Alert box -----------------------------
Procedure Entry_error
  @Entry_error_sound
  Alert 3," Entry error,  try again!",1," OK ",A
  Clr A
Return
' ----------------------------- Error sound -------------------------------
Procedure Entry_error_sound
  Sound 1,12,12,4
  Pause 3
  Sound 1,0,0,0
Return
' ---------------------------- Force addressee check --------------------
Procedure F_check
  If F_name$=""        !## Ensures an addressee for Force commands.
    Mtxt$=" This command requires an addressee (unit)."
    @Entry_error_sound
    Alert 1,Mtxt$,1," OK ",A
    @F_entry
  Endif
  If Not Instr(T$,F_name$) !## To precede Force cmds by "FOR 'addressee' "
    T$=F_name$+" "+T$
  Endif
  Clr A
  Print At(38-Int(Len(F_name$)/2),19);"  "+F_name$+"  ";
Return
' ---------------------------- Force addressee entry --------------------
Procedure F_entry
  F_name$="FOR "
  Print At(22,Ytext%);"Enter name of addressee (unit): ";
  Form Input 5,Fi$
  F_name$=F_name$+Upper$(Fi$)
  @Cclear_middle
```

```
  Print At(38-Int(Len(F_name$)/2),19);"  "+F_name$+"  ";
Return
' ---------------------------- Latitude entry check ----------------------
Procedure Lat_check
  Cx%=Crscol
  Cy%=Crslin
  Form Input 3,Lat$
  Ltr=Asc(Right$(Lat$))  !## ASCII value of direction (78,83,110,115)(NSns)
  If Val?(Lat$)>0        !## is there a digit in the string?
    If Val(Lat$)>=0 And Val(Lat$)<90   !## Is number value betwn 0 and 90?
      If Ltr=78 Or Ltr=83 Or Ltr=110 Or Ltr=115   !## correct direction?
        Goto Last_lat
      Endif
    Endif
  Endif
  @Entry_error
  Print At(Cx%,Cy%);"          ";      !## clear cursor area
  Print At(Cx%,Cy%);                   !## reposition cursor
  @Lat_check                           !## check it again
  Last_lat:                            !## latitude OK; continue.
Return
' ---------------------------- Longitude entry check ----------------------
Procedure Long_check
  Cx%=Crscol
  Cy%=Crslin
  Form Input 4,Long$
  Ltr=Asc(Right$(Long$)) !## ASCII value of direction (69,87,101,119)(EWew)
  If Val?(Long$)>0       !## is there a digit in the string?
    If Val(Long$)>=0 And Val(Long$)<181 !## Is number value betwn 0 and 180?
      If Ltr=69 Or Ltr=87 Or Ltr=101 Or Ltr=119   !## correct direction?
        Goto Last_long
      Endif
    Endif
  Endif
  @Entry_error
  Print At(Cx%,Cy%);"          ";      !## clear cursor area
  Print At(Cx%,Cy%);                   !## reposition cursor
  @Long_check                          !## check it again
  Last_long:                           !## longitude OK; continue.
Return
'
' ---------------------- Pure number entry and checking -------------------
'                      !## need lolim%, hilim%, numlen% as input;
'                      !## gives Pnum$ as output.
Procedure Number_entry
  Cx%=Crscol    ! remember x posit of cursor
  Cy%=Crslin    ! remember y posit of cursor
  Form Input Numlen%,Pnum$
  '           Val(Pnum$)  = actual numeric value of string Pnum$.
  '           Val?(Pnum$) = number of characters in Pnum$ that are numeric.
  '           Len(Pnum$)  = length of string Pnum$.
  If Val(Pnum$)<Lolim% Or Val(Pnum$)>Hilim% Or Val?(Pnum$)<>Len(Pnum$)
```

```
    Print At(Cx%,Cy%);"     ";
    Print At(Cx%,Cy%);
    @Entry_error
    @Number_entry
  Endif
Return
'
' ---------------- Common code for Aircraft MISSION menu selections -------
'
Procedure Mission_common
  Print At(40-Int(Len(Cstr2$)/2),Ytext%);Cstr2$;     !## 'Mission' string
  Inc Ytext%
  T$=T$+Cstr2$
  @Tc_choice
  '
  If Not Aclaunch!   !## not using Launch sequence
    @Show_cmd
  Endif
Return
'
'                        allows choice between TIME & continue.
Procedure Tc_choice
  @Show_cmd
  Alert 2,"   select ... ",2,"TIME continue",C
  If C=1
    @Time_entry
  Endif
  Clr C
Return
'
'                             Common minute entry
Procedure Minute_entry
  Inc Ytext%
  Inc Ytext%
  Print At(28,Ytext%);"Enter minutes (1-9999)? ";
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+Pnum$
Return
'
'                                Maneuvers sub-sub-proc
Procedure Time_entry
  Print At(26,Ytext%);"Enter start minute (1-999): ";
  Lolim%=1
  Hilim%=999
  Numlen%=3
  @Number_entry
  T$=T$+" TIME "+Pnum$
  Ytext%=Crslin
Return
```

```
'
'                                Maneuvers sub-sub-proc
Procedure Course_entry
  Print At(26,Ytext%);"Enter course (0-359  True): ";
  Lolim%=0
  Hilim%=359
  Numlen%=3
  @Number_entry
  T$=T$+Pnum$
  Ytext%=Crslin
Return
'
'                                Maneuvers sub-sub-proc
Procedure Bearing_entry
  Print At(26,Ytext%);"Enter bearing (0-359  True): ";
  Lolim%=0
  Hilim%=359
  Numlen%=3
  @Number_entry
  T$=T$+Pnum$
  Ytext%=Crslin
Return
'
'                               Maneuvers sub-sub-proc
Procedure Speed_entry
  Print At(27,Ytext%);"Enter speed (1-9999 kts): ";
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+" "+Pnum$
  Ytext%=Crslin
Return
'
'                               Maneuvers sub-sub-proc
Procedure Distance_entry
  Print At(21,Ytext%);"Enter distance or range (1-9999 nmi): ";
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+" "+Pnum$
  Ytext%=Crslin
Return
'
'                                    sub-sub proc
Procedure Altitude_entry
  Cstr$="Enter altitude (1-90,000 ft): "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Cstr$=""
  Lolim%=1
  Hilim%=90000
```

```
  Numlen%=5
  @Number_entry
  T$=T$+Pnum$
  Ytext%=Crslin
Return
'
'
Procedure Name_time_choice
  @Show_cmd
  Alert 2,"        select ...       ",3,"<name> TIME continue",A
  If A=1
    @Cclear_middle
    Cstr$="Enter equipment name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
    Ytext%=Crslin
    If Bar$(Menu(0))=" Emitter"
      @Name_time_choice
    Else
      @Tc_choice  !## Time/continue choice
    Endif
  Endif
  '
  If A=2
    @Time_entry
  Endif
  Clr A   !## don't clear A
Return
'
'
Procedure Use_time_choice
  @Show_cmd
  Alert 2,"        select ...       ",3,"USING TIME continue",A
  '
  If A=1
    Print At(26,Ytext%);"Using what equipment? : ";
    Form Input 5,Fi$
    T$=T$+" USING "+Upper$(Fi$)
    Ytext%=Crslin
    '
    @Tc_choice  !## Time/continue choice
    '
  Endif
  If A=2
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                                  Track common entry
```

```
'   !## format is XX####; ie. AS234
'   !## for letters, gives Chr$(B) as output of last character.
'   !## for numbers, need lolim%, hilim%, numlen% as input; Pnum$ is output
'
Procedure Track_entry
  Cx%=Crscol    ! remember x posit of cursor
  Cy%=Crslin    ! remember y posit of cursor
  N%=Cx%        ! set baseline number; allows certain # of iterations
  '
  ' - - - - -  character portion of track number. - - - - -
  Te:
  B=Asc(Upper$(Chr$(Inp(2))))
  If B<65 Or B>90      !## if char is not A - Z, then...
    Print At(Cx%,Cy%);" ";
    Print At(Cx%,Cy%);
    @Entry_error       !## alert box & sound
    Goto Te            !## recursion
  Endif
  Print Chr$(B);
  T$=T$+Chr$(B)
  Inc Cx%
  Print At(Cx%,Cy%);
  If N%=Cx%-1     !## allows only one use of this IF loop
    Goto Te
  Endif
  ' - - - - - - - - numeral portion of track number - - - - -
  Lolim%=0
  Hilim%=999
  Numlen%=3
  @Number_entry
  T$=T$+Pnum$
Return   !@Track_entry
'
'                              Comms text entry
Procedure Text_entry  !## need to enter with I%= to number of text lines.
  @Show_cmd
  @Cclear_middle
  N%=1
  Ccomtxt2:
  Mtxt$="   Enter text?    (max "+Str$(I%)+" lines) "
  Alert 2,Mtxt$,0,"  Yes   No (BT)",A
  If A=1
    Print At(2,N%+3);Str$(N%)+": ";
    Form Input 75,Fi$
    T$=T$+Upper$(Fi$)+" "
    Inc N%
    If N%<I%+1   !## allows only I% lines of text
      Goto Ccomtxt2
    Endif
  Endif
  Clr A
  T$=T$+"BT"
```

```
Return
'
'                  allows completion of command string using same menu
Procedure Partial
  Menu Off   !## displays menu bar in "normal" mode
  @Show_cmd
  Do
    On Menu
  Loop
Return
'
' ##############################################################################
'
'       COMMAND PROCEDURES
'
' ##############################################################################
'
'                                                ASTAB sub-proc
Procedure Bbearing
  Brng!=False
  Mtxt$="    Bearing & Range FROM...  "
  '
  Brng:
  Alert 2,Mtxt$,0,"Force Position Track",A
  '
  If A=1 Then
    T$=T$+"FORCE "
    @Show_cmd
    Cstr$="Enter Force name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Endif
  '
  If A=2 Then
    T$=T$+"POSITION "
    Print At(20,11);Lat_str$;
    @Lat_check
    Print At(20,12);Long_str$;
    @Long_check
    T$=T$+Upper$(Lat$)+" "+Upper$(Long$)
  Endif
  '
  If A=3
    T$=T$+"TRACK "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  Clr A
  @Show_cmd
```

```
  '
  If Not Brng!  !## allows ONE loop in this procedure.
    Mtxt$="    Bearing & Range TO...  "
    Brng!=True
    @Cclear_middle
    T$=T$+" "
    Goto Brng
  Endif
Return
'
'                                          ASTAB sub-proc
Procedure Ccpa
  Cpa!=False
  Mtxt$="     CPA of...  "
  '
  Cpa:
  Alert 2,Mtxt$,0,"Force  Track ",A
  '
  If A=1 Then
    T$=T$+"FORCE "
    @Show_cmd
    Cstr$="Enter Force name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Endif
  '
  If A=2
    T$=T$+"TRACK "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  Clr A
  '
  If Not Cpa!  !## allows ONE loop in this procedure.
    Mtxt$="     CPA to...   "
    Cpa!=True
    @Show_cmd
    @Cclear_middle
    T$=T$+" "
    Goto Cpa
  Endif
Return
'
'                                          ASTAB sub-proc
Procedure Cclassify
  @Show_cmd
  Cstr$="Enter track number: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  @Track_entry
```

```
  '
  @Cclear_middle
  @Show_cmd
  Cstr$="Enter classification: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
Return
'
'                                          ASTAB sub-proc
Procedure Ddrop
  Alert 2,"     DROP track...     ",0,"Old Range Track # ",A
  '
  If A=1 Then
    T$=T$+"OLD "
    @Show_cmd
    Cstr$="(range); enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
    T$=T$+" "
    @Show_cmd
    @Cclear_middle
    Cstr$="(thru); enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If A=2 Then
    T$=T$+"RANGE "
    @Show_cmd
    Cstr$="Enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
    T$=T$+" "
    @Show_cmd
    @Cclear_middle
    Cstr$="(thru); enter track number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If A=3
    Mtxt$="   Drop a track?  "
    Ddrop:
    Alert 2,Mtxt$,0,"  Yes   No",B
    If B=1
      Cstr$="Enter track number: "
      Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
      @Track_entry
      Mtxt$="   Drop another track?  "
      @Show_cmd
      T$=T$+" "
```

```
        Goto Ddrop
      Endif
    Endif
    Clr A,B
  Return
  '
  '                                              ASTAB sub-proc
  Procedure Pprint
    Alert 2,"    Print...    ",0," ASTAB  Plot",A
    If A=1
      T$=T$+"ASTAB "
      Alert 2,"    Select...   ",0,"ASTAB #  All",B
      If B=1
        Cstr$="Enter ASTAB number: "
        Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
        Lolim%=0
        Hilim%=6
        Numlen%=1
        @Number_entry
        T$=T$+Pnum$
      Endif
      If B=2
        T$=T$+"ALL"
      Endif
    Endif
    Clr B
    '
    If A=2
      T$=T$+"PLOT "
      Alert 2,"    Plot...   ",0,"Interval continue",B
      If B=1
        T$=T$+"INTERVAL "
        @Minute_entry
      Endif
    Endif
    Clr A,B
  Return
  '
  '                                Graphics sub-proc
  Procedure Ccenter
    Mtxt$="         CENTER                    ( plot at ... )"
    Alert 2,Mtxt$,0,"FORCE POSITION TRACK",A
    If A=1 Then
      T$=T$+"FORCE"
      Print At(30,8);T$;
      Print At(17,10);"Enter name of force to be centered: ";
      Form Input 5,Fi$
      T$=T$+" "+Upper$(Fi$)
    Endif
    If A=2 Then
      T$=T$+"POSITION"
      Print At(30,9);"Center plot at ..."
```

```
    Print At(20,11);Lat_str$;
    @Lat_check
    Print At(20,12);Long_str$;
    @Long_check
    T$=T$+" "+Upper$(Lat$)+" "+Upper$(Long$)
  Endif
  If A=3 Then
    T$=T$+"TRACK "
    Cstr$=T$+"at (track number): "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  Clr A
Return
'
'                                    Graphics sub-proc
Procedure Rradius
  Print At(35,8);T$
  Print At(20,10);"Enter radius of plot (1-9999 nmi): ";
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+Pnum$
Return
'
'                                   Graphics sub-proc
Procedure Sshift
  Print At(35,6);T$
  '
  Print At(10,8);"Enter distance to shift center of plot (1-9999 nmi): ";
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+Pnum$
  '
  Print At(15,9);"Enter direction of shift (0-359  True): ";
  Lolim%=0
  Hilim%=359
  Numlen%=3
  @Number_entry
  T$=T$+" "+Pnum$
Return
'
'                                   Graphics sub-proc
Procedure Llabel
  @Label("",*Label_return$)
  If Instr(Label_return$,"al_lab")>0
    T$=T$+"ALL"
  Endif
  If Instr(Label_return$,"large_lab")>0
```

```
    T$=T$+"LARGE"
  Endif
  If Instr(Label_return$,"small_lab")>0
    T$=T$+"SMALL"
  Endif
  If Instr(Label_return$,"off_lab")>0
    T$=T$+"OFF"
  Endif
Return
'
'                                          Label DIOX
Procedure Label(Preselect$,Postselect)
  Hidem
  Local
Screen$,Temp$,Xm,Ym,Button$,Radio1_old$,Radio2_old$,Radio3_old$,Radio4_old$,-
Radio5_old$,Radio1_new$,Radio2_new$,Radio3_new$,Radio4_new$,Radio5_new$,Stat_exi

  Sget Screen$
  Print At(36,6);T$
  Local All_lab_stat$,All_lab_stat
  Local Large_lab_stat$,Large_lab_stat
  Local Small_lab_stat$,Small_lab_stat
  Local Off_lab_stat$,Off_lab_stat
  @Drawshapes_label
  @All_lab(0)
  @Large_lab(0)
  @Small_lab(0)
  @Off_lab(0)
  Showm
  Do
    If Mousek=1
      Mouse Xm,Ym,Void
      @Find_button_label(Xm,Ym,*Button$)
      If Button$="al_lab"
        If All_lab_stat
          @All_lab(0)
          Let All_lab_stat=False
          Let All_lab_stat$=""
          Let Stat_exit=False
        Else
          @All_lab(-1)
          Let All_lab_stat=True
          Let All_lab_stat$="al_lab"
          Let Stat_exit=True
        Endif
      Endif
      If Button$="large_lab"
        If Large_lab_stat
          @Large_lab(0)
          Let Large_lab_stat=False
          Let Large_lab_stat$=""
          Let Stat_exit=False
        Else
```

```
        @Large_lab(-1)
        Let Large_lab_stat=True
        Let Large_lab_stat$="large_lab"
        Let Stat_exit=True
      Endif
    Endif
    If Button$="small_lab"
      If Small_lab_stat
        @Small_lab(0)
        Let Small_lab_stat=False
        Let Small_lab_stat$=""
        Let Stat_exit=False
      Else
        @Small_lab(-1)
        Let Small_lab_stat=True
        Let Small_lab_stat$="small_lab"
        Let Stat_exit=True
      Endif
    Endif
    If Button$="off_lab"
      If Off_lab_stat
        @Off_lab(0)
        Let Off_lab_stat=False
        Let Off_lab_stat$=""
        Let Stat_exit=False
      Else
        @Off_lab(-1)
        Let Off_lab_stat=True
        Let Off_lab_stat$="off_lab"
        Let Stat_exit=True
      Endif
    Endif
  Endif
  Exit If False
  Exit If (Button$="al_lab")
  Exit If (Button$="large_lab")
  Exit If (Button$="small_lab")
  Exit If (Button$="off_lab")
  Pause 4
Loop
If Stat_exit
  Temp$=Temp$+"("+All_lab_stat$+")"
  Temp$=Temp$+"("+Large_lab_stat$+")"
  Temp$=Temp$+"("+Small_lab_stat$+")"
  Temp$=Temp$+"("+Off_lab_stat$+")"
  *Postselect=Temp$
Endif
Do
  @Find_button_label(Xm,Ym,*Button$)
  Exit If Stat_exit
Loop
Pause 7
```

```
  Sput Screen$
Return
Procedure Drawshapes_label
  Box 160,86,451,30
Return
Procedure All_lab(Selected)
  @Draw_text_in_box("ALL",190,74,0,1,6,12,1,2,Selected)
Return
Procedure Large_lab(Selected)
  @Draw_text_in_box("LARGE",251,74,0,1,6,4,1,2,Selected)
Return
Procedure Small_lab(Selected)
  @Draw_text_in_box("SMALL",320,74,0,1,6,4,1,2,Selected)
Return
Procedure Off_lab(Selected)
  @Draw_text_in_box("OFF",397,74,0,1,6,12,1,2,Selected)
Return
Procedure Find_button_label(X_mouse,Y_mouse,Button_selected)
  If (X_mouse>176) And (X_mouse<228) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="al_lab"
    Goto Found_button_label
  Endif
  If (X_mouse>245) And (X_mouse<297) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="large_lab"
    Goto Found_button_label
  Endif
  If (X_mouse>314) And (X_mouse<366) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="small_lab"
    Goto Found_button_label
  Endif
  If (X_mouse>383) And (X_mouse<435) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="off_lab"
    Goto Found_button_label
  Endif
  *Button_selected=" "
  Found_button_label:
Return
'
'                                    Graphics sub-proc
Procedure Llob
  Mtxt$="       LOB       "
  Alert 2,Mtxt$,0,"ALL FORCE OFF",A
  If A=1 Then
    T$=T$+"ALL"
  Endif
  If A=2 Then
    T$=T$+"FORCE"
  Endif
  If A=3 Then
    T$=T$+"OFF"
  Endif
  Clr A
```

```
Return
'
'                                    Graphics sub-proc
Procedure Mmark_track
  Mtxt$="     MARK TRACK        "
  Alert 2,Mtxt$,0,"ENEMY FRIENDLY NEUTRAL",A
  If A=1 Then
    T$=T$+"ENEMY "
  Endif
  If A=2 Then
    T$=T$+"FRIENDLY "
  Endif
  If A=3 Then
    T$=T$+"NEUTRAL "
  Endif
  T1$="   "+T$
  Clr A
  '
  Alert 2,T1$,0,"AIR SUB SURFACE",A
  If A=1 Then
    T$=T$+"AIR "
  Endif
  If A=2 Then
    T$=T$+"SUB "
  Endif
  If A=3 Then
    T$=T$+"SURFACE "
  Endif
  Cstr$=T$+" (at position...)"
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  Print At(27,Ytext%);"Enter unit's latitude: ";
  @Lat_check
  Inc Ytext%
  Print At(27,Ytext%);"Enter unit's longitude: ";
  @Long_check
  T$=T$+Upper$(Lat$)+" "+Upper$(Long$)
  Inc Ytext%
  Print At(23,Ytext%);"Choose name for this track: ";
  Form Input 5,Fi$
  Inc Ytext%
  Print At(17,Ytext%);"Choose second name for this track (optional): ";
  Form Input 5,Mtk_name2$
  T$=T$+" "+Upper$(Fi$)+" "+Upper$(Mtk_name2$)
  Clr A
Return
'
'                                  Graphics sub-proc
Procedure Mmark_bearing
  Print At(35,Ytext%);T$
  Inc Ytext%
```

```
  Inc Ytext%
  Print At(25,10);"Choose name for this bearing: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)+" "
  Inc Ytext%
  @Bearing_entry
Return
'
'                                    Graphics sub-proc
Procedure Uunmark_track
  Print At(30,8);T$
  Print At(15,10);"Enter (first) name of track to unmark: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
Return
'
'                                     Graphics sub-proc
Procedure Pplace
  Mtxt$="     PLACE       "
  Alert 2,Mtxt$,0,"XMARK CIRCLE GRID",A
  If A=1 Then
    T$=T$+"XMARK "
  Endif
  If A=2 Then
    T$=T$+"CIRCLE "
  Endif
  If A=3 Then
    T$=T$+"GRID "
  Endif
  Clr A
  '
  Mtxt$="   "+T$+"        on              "
  Alert 2,Mtxt$,0,"FORCE POSITION TRACK",A
  If A=1 Then
    T$=T$+"FORCE "
  Endif
  If A=2 Then
    T$=T$+"POSITION "
  Endif
  If A=3 Then
    T$=T$+"TRACK "
  Endif
  Clr A
Return
'
'                                    Graphics sub-proc
Procedure Ppim
  Mtxt$="      PIM        "
  Alert 2,Mtxt$,0,"DEFINE ADD CHANGE",A
  If A=1 Then
    T$=T$+"DEFINE"
  Endif
```

```
  If A=2 Then
    T$=T$+"ADD"
  Endif
  If A=3 Then
    T$=T$+"CHANGE"
  Endif
  Clr A
Return
'
'                                player COMMS sub-proc
Procedure Iintell
  Alert 2," Intelligence report for...",0,"Blue  Orange ",A
  If A=1 Then
    T$=T$+"BLUE"
  Endif
  If A=2 Then
    T$=T$+"ORANGE"
  Endif
  Clr A
  '
  Get 0,140,639,162,Canex_box$
  Alert 2,"        Select...      ",0,"View Time continue",A
  If A=1 Then
    @Show_cmd
    Cstr$="Enter view number: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)+" "
    I%=20
    @Text_entry
  Endif
  If A=2 Then
    @Time_entry
    I%=20
    @Text_entry
    T$=T$+" "
  Endif
  Clr A
  Put 0,140,Canex_box$
Return
'
'                                  player COMMS sub-proc
Procedure Mmessage
  Alert 2,"    Message to...  ",0,"Blue  Orange ",A
  If A=1
    T$=T$+"BLUE "
  Endif
  If A=2
    T$=T$+"ORANGE "
  Endif
  Clr A
  '
```

```
  @Show_cmd
  Cstr$="Enter view number: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)+" "
  I%=8
  @Text_entry
Return
'
'                                        Umpire sub-proc
Procedure Ppause
  Mtxt$="      PAUSE Pause: game pauses. "
  Mtxt$=Mtxt$+"Lock: on stations. "
  Mtxt$=Mtxt$+"Refresh: of remote db."
  Alert 2,Mtxt$,0,"PAUSE LOCK REFRESH",A
  If A=2 Then
    T$=T$+"LOCK "
  Endif
  If A=3 Then
    T$=T$+"REFRESH "
  Endif
  Clr A
Return
'
'                                     Umpire sub-proc
Procedure Eend
  Mtxt$="       END End: ends game. "
  Mtxt$=Mtxt$+"No: no auto logout. "
  Mtxt$=Mtxt$+"Yes: with auto logout."
  Alert 2,Mtxt$,0,"END NO YES",A
  If A=2 Then
    T$=T$+"NO "
  Endif
  If A=3 Then
    T$=T$+"YES "
  Endif
  Clr A
Return
'
'                                   Umpire sub-proc
Procedure Ccopy
  @Copy("",*Copy_return$)
  If Instr(Copy_return$,"all_cop")>0
    T$=T$+"ALL"
  Endif
  If Instr(Copy_return$,"blue_cop")>0
    T$=T$+"BLUE"
  Endif
  If Instr(Copy_return$,"orange_cop")>0
    T$=T$+"ORANGE"
  Endif
  If Instr(Copy_return$,"off_cop")>0
```

```
    T$=T$+"OFF"
  Endif
Return
'
'                                          Copy DIOX
Procedure Copy(Preselect$,Postselect)
  Hidem
  Local
Screen$,Temp$,Xm,Ym,Button$,Radio1_old$,Radio2_old$,Radio3_old$,Radio4_old$,-
Radio5_old$,Radio1_new$,Radio2_new$,Radio3_new$,Radio4_new$,Radio5_new$,Stat_exi

  Sget Screen$
  Print At(38,6);T$
  Local All_cop_stat$,All_cop_stat
  Local Blue_cop_stat$,Blue_cop_stat
  Local Orange_cop_stat$,Orange_cop_stat
  Local Off_cop_stat$,Off_cop_stat
  @Drawshapes_copy
  @All_cop(0)
  @Blue_cop(0)
  @Orange_cop(0)
  @Off_cop(0)
  Showm
  Do
    If Mousek=1
      Mouse Xm,Ym,Void
      @Find_button_copy(Xm,Ym,*Button$)
      If Button$="all_cop"
        If All_cop_stat
          @All_cop(0)
          Let All_cop_stat=False
          Let All_cop_stat$=""
          Let Stat_exit=False
        Else
          @All_cop(-1)
          Let All_cop_stat=True
          Let All_cop_stat$="all_cop"
          Let Stat_exit=True
        Endif
      Endif
      If Button$="blue_cop"
        If Blue_cop_stat
          @Blue_cop(0)
          Let Blue_cop_stat=False
          Let Blue_cop_stat$=""
          Let Stat_exit=False
        Else
          @Blue_cop(-1)
          Let Blue_cop_stat=True
          Let Blue_cop_stat$="blue_cop"
          Let Stat_exit=True
        Endif
      Endif
```

```
      If Button$="orange_cop"
        If Orange_cop_stat
          @Orange_cop(0)
          Let Orange_cop_stat=False
          Let Orange_cop_stat$=""
          Let Stat_exit=False
        Else
          @Orange_cop(-1)
          Let Orange_cop_stat=True
          Let Orange_cop_stat$="orange_cop"
          Let Stat_exit=True
        Endif
      Endif
      If Button$="off_cop"
        If Off_cop_stat
          @Off_cop(0)
          Let Off_cop_stat=False
          Let Off_cop_stat$=""
          Let Stat_exit=False
        Else
          @Off_cop(-1)
          Let Off_cop_stat=True
          Let Off_cop_stat$="off_cop"
          Let Stat_exit=True
        Endif
      Endif
    Endif
    Exit If False
    Exit If (Button$="all_cop")
    Exit If (Button$="blue_cop")
    Exit If (Button$="orange_cop")
    Exit If (Button$="off_cop")
    Pause 4
  Loop
  If Stat_exit
    Temp$=Temp$+"("+All_cop_stat$+")"
    Temp$=Temp$+"("+Blue_cop_stat$+")"
    Temp$=Temp$+"("+Orange_cop_stat$+")"
    Temp$=Temp$+"("+Off_cop_stat$+")"
    *Postselect=Temp$
  Endif
  Do
    @Find_button_copy(Xm,Ym,*Button$)
    Exit If Stat_exit
  Loop
  Pause 7
  Sput Screen$
Return
Procedure Drawshapes_copy
  Box 149,86,471,30
Return
Procedure All_cop(Selected)
```

```
  @Draw_text_in_box("ALL",183,74,0,1,6,16,1,2,Selected)
Return
Procedure Blue_cop(Selected)
  @Draw_text_in_box("BLUE",256,74,0,1,6,12,1,2,Selected)
Return
Procedure Orange_cop(Selected)
  @Draw_text_in_box("ORANGE",324,74,0,1,6,4,1,2,Selected)
Return
Procedure Off_cop(Selected)
  @Draw_text_in_box("OFF",413,74,0,1,6,16,1,2,Selected)
Return
Procedure Find_button_copy(X_mouse,Y_mouse,Button_selected)
  If (X_mouse>165) And (X_mouse<225) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="all_cop"
    Goto Found_button_copy
  Endif
  If (X_mouse>242) And (X_mouse<302) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="blue_cop"
    Goto Found_button_copy
  Endif
  If (X_mouse>318) And (X_mouse<378) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="orange_cop"
    Goto Found_button_copy
  Endif
  If (X_mouse>395) And (X_mouse<455) And (Y_mouse>64) And (Y_mouse<78)
    *Button_selected="off_cop"
    Goto Found_button_copy
  Endif
  *Button_selected=" "
  Found_button_copy:
Return
'
'                                  Umpire sub-proc
Procedure Rrelocate
  Print At(35,8);T$
  Print At(20,10);"Enter name of unit to be relocated: ";
  Form Input 5,Fi$
  '
  Print At(20,11);Lat_str$;
  @Lat_check
  '
  Print At(20,12);Long_str$;
  @Long_check
  '
  T$=T$+Upper$(Fi$)+" "+Upper$(Lat$)+" "+Upper$(Long$)
  Mtxt$="RELOCATE relative to another unit?"
  Alert 2,Mtxt$,0," YES  NO",A
  If A=1 Then
    Print At(20,14);"Relative to which unit? ";
    Form Input 5,Fi$
    T$=T$+" RELATIVE "+Upper$(Fi$)
  Endif
```

```
  '
  Clr A
Return
'
'                                   Umpire sub-proc
Procedure Ttime
  Print At(37,8);T$
  Print At(15,10);"Enter time of game minute (10-400 seconds): ";
  Lolim%=10
  Hilim%=400
  Numlen%=3
  @Number_entry
  T$=T$+Pnum$
Return
'
'                                   Umpire sub-proc
Procedure Sset
  Mtxt$="           SET Fast: no output until... "
  Mtxt$=Mtxt$+"Normal: continuous messages. "
  Mtxt$=Mtxt$+"Zulu: time change."
  Alert 2,Mtxt$,0,"FAST NORMAL ZULU",A
  If A=1 Then
    T$=T$+"FAST "
    @Sset_fast
  Endif
  If A=2 Then
    T$=T$+"NORMAL "
  Endif
  If A=3 Then
    T$=T$+"ZULU "
    @Sset_zulu
  Endif
  Clr A
Return
'
'                                   Umpire sub-proc
Procedure Sset_fast
  Print At(33,8);T$
  Print At(12,10);"Enter game minute when output should start (1-999): ";
  Lolim%=1
  Hilim%=999
  Numlen%=3
  @Number_entry
  T$=T$+Pnum$
Return
'
'                                   Umpire sub-proc
Procedure Sset_zulu
  Mtxt$="   SET ZULU   "
  Alert 2,Mtxt$,0," AHEAD   BACK ",A
  If A=1 Then
    T$=T$+"AHEAD "
```

```
    Else
      T$=T$+"BACK "
    Endif
    Print At(30,8);T$
    '
    Print At(25,10);"Enter hours (0-23): ";     !## hour entry
    Lolim%=0
    Hilim%=23
    Numlen%=2
    @Number_entry
    T$=T$+Pnum$
    '
    Print At(25,11);"Enter minutes (0-59): ";     !## minute entry
    Lolim%=0
    Hilim%=59
    Numlen%=2
    @Number_entry
    T$=T$+" "+Pnum$
    '
    Clr A
  Return
  '
  '                                    Umpire sub-proc
  Procedure Enable_disable
    @F_check
    Print At(30,Ytext%);T$
    '
    If Instr(T$," DISABLE ")
      @Minute_entry
      T$=T$+" "
    Endif
    '
    Print At(25,Ytext%+2);"Enter equipment name: ";
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Return
  '
  '                                    Umpire sub-proc
  Procedure Expend_replenish
    @F_check
    Print At(30,8);T$
    Print At(25,10);"Enter amount (number): ";
    Lolim%=1
    Hilim%=999
    Numlen%=3
    @Number_entry
    T$=T$+Pnum$+" "
    '
    Print At(25,11);"Enter equipment name: ";
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Return
```

```
'
'
'
' ############ Second level Procedures for FORCE menu 'A' items #############
'
'                                Maneuvers sub-proc
Procedure Ccourse
  Print At(33,Ytext%);T$
  Ytext%=Ytext%+2
  @Course_entry
  Inc Ytext%
  '
  Mtxt$="  Enter Course change... "   !## for alert box text
  Alert 2,Mtxt$,1,"NOW  LATER ",A
  If A=2 Then
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                                Maneuvers sub-proc
Procedure Sspeed
  Print At(33,Ytext%);T$
  Ytext%=Ytext%+2
  @Speed_entry
  Inc Ytext%
  '
  Mtxt$="  Enter Speed change... "   !## for alert box text
  Alert 2,Mtxt$,1,"NOW  LATER ",A
  If A=2 Then
    @Time_entry
  Endif
  Clr A
Return
'
'                                Maneuvers sub-proc
Procedure Pproceed
  Alert 2," "+T$+"...",0,"COURSE POSITION",A
  Print At(33,Ytext%);T$
  Ytext%=Ytext%+2
  '
  If A=1
    @Course_entry
    @Distance_entry
  Endif
  If A=2
    Print At(23,Ytext%);Lat_str$;
    @Lat_check
    Inc Ytext%
    Print At(23,Ytext%);Long_str$;
    @Long_check
```

```
    Inc Ytext%
    T$=T$+Upper$(Lat$)+" "+Upper$(Long$)
  Endif
  Clr A
  '
  Alert 2,"        select ...       ",3,"SPEED TIME continue",A
  '
  If A=1
    @Speed_entry
    @Tc_choice
  Endif
  '
  If A=2
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                              Maneuvers sub-proc
Procedure Sstation
  Print At(33,Ytext%);T$
  Ytext%=Ytext%+2
  @Bearing_entry
  Print At(33,Ytext%+1);"FROM (guide) ..."
  Ytext%=Crslin
  Print At(22,Ytext%);"Enter name of addressee (unit): ";
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
  @Distance_entry
  @Tc_choice
Return
'
'                              Maneuvers sub-proc
Procedure Ssearch
  @Show_cmd
  Print At(28,Ytext%);"Enter name of plan: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  Ytext%=Crslin
  @Tc_choice
Return
'
'                              Maneuvers sub-proc
Procedure Uuse
  Print At(33,Ytext%);T$
  Inc Ytext%
  Inc Ytext%
  Print At(28,Ytext%);"Enter name of plan: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  Ytext%=Crslin
```

```
  '
  @Use_time_choice
Return
'
'                                Maneuvers sub-proc
Procedure Eexecute
  Print At(33,Ytext%);T$
  Inc Ytext%
  Inc Ytext%
  Print At(22,Ytext%);"Enter name of contingency plan: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  Ytext%=Crslin
  @Tc_choice
Return
'
'                                                    Sensors sub-proc
'                  Activate/Silence sub-sub procedure
'                Note: Activate & Silence have own menus.
'
Procedure Eemitter
  @Show_cmd
  Alert 2,"        select ...       ",3,"TIME <name> continue",A
  If A=1
    @Time_entry
  Endif
  If A=2
    Print At(27,Ytext%);"Enter name of emitter: ";
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
  Endif
  Clr A
Return
'
'                                 Activate/Silence sub-sub procedure
Procedure Ssurvsat
  @Show_cmd
  Mtxt$="   "+T$+"   "
  '
  Print At(25,Ytext%);"Enter name of Survsat: ";
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  Inc Ytext%
  Print At(28,Ytext%);"Enter force name: ";
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
  '
  @Cclear_middle
  @Show_cmd
  Alert 2,Mtxt$,0,"ORBITAL STATNARY",A
  If A=1
    T$=T$+" ORBITAL "
```

```
    @Show_cmd
    Cstr$="From latitude: "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Lat_check
    T$=T$+Upper$(Lat$)
    Cstr$=Cstr$+Upper$(Lat$)+"   To latitude: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Lat_check
    T$=T$+" "+Upper$(Lat$)
  Endif
  '
  If A=2
    T$=T$+" STATIONARY"
    @Show_cmd
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                                                    Sensors sub-proc
Procedure Jjam
  @Show_cmd
  Print At(28,Ytext%);"Enter radar name: ";
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
Return
'
'                                                    Sensors sub-proc
Procedure Ccease
  Cstr$="Cease jamming radar"
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  Alert 2,"        select ...       ",3,"TIME name continue",A
  If A=1
    @Time_entry
  Endif
  If A=2
    Print At(27,Ytext%);"Enter name of radar: ";
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
  Endif
  Clr A
Return
'
'                                                    Sensors sub-proc
Procedure Eemcon
  @Show_cmd
  Print At(28,Ytext%);"Enter plan name: ";
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
```

```
  '
  Mtxt$="   EXEMPT any units? "
  Emc:
  @Cclear_middle
  Alert 2,Mtxt$,0,"  Yes   No",A
  If A=1
    T$=T$+" EXEMPT"
    Cstr$="EXEMPT which unit?: "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
    Mtxt$="   EXEMPT more units? "
    @Show_cmd
    Goto Emc
  Endif
  '
  Clr A
Return
'
'                                  Engage sub-proc
Procedure Ffire
  @Show_cmd
  Print At(19,Ytext%);"Enter number of weapons to fire (1-99): ";
  Lolim%=1
  Hilim%=99
  Numlen%=2
  @Number_entry
  T$=T$+Pnum$
  Ytext%=Crslin
  '
  Print At(15,Ytext%);"Enter name of weapon; TLAM, MK48, HRPON, etc: ";
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)+" "
  Ytext%=Crslin
  '
  @Cclear_middle
  @Show_cmd
  Mtxt$="   "+T$+"   "
  Alert 2,Mtxt$,0,"NUCLEAR CRUISE TORPEDO",A
  If A=1
    T$=T$+"NUCLEAR "
    @Show_cmd
    Mtxt$=Mtxt$+"NUCLEAR "
    Alert 2,Mtxt$,0,"CRUISE TORPEDO",B
  Endif
  '
  If A=2 Or B=1
    T$=T$+"CRUISE "
    @Show_cmd
    Mtxt$=Mtxt$+"CRUISE missiles "
    Alert 2,Mtxt$,0,"AT BEARING",C
    '
```

```
      If C=1
        T$=T$+"AT "
        Cstr$="at which shorebase? "
        Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
        Form Input 5,Fi$
        T$=T$+Upper$(Fi$)
        Ytext%=Crslin
        @Show_cmd
      Endif
      '
      If C=2
        T$=T$+"BEARING "
        Ytext%=Ytext%+2
        @Bearing_entry
        @Show_cmd
        Mtxt$=Mtxt$+"   BEARING "+Pnum$
        Alert 2,Mtxt$,0," DELAY   RANGE ",D
        @Cclear_middle
        '
        If D=1
          T$=T$+" DELAY "
          Cstr$="delay... "
          Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
          @Minute_entry
          @Show_cmd
        Endif
        '
        If D=2
          T$=T$+" RANGE"
          Cstr$="range..."
          Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
          Ytext%=Ytext%+2
          @Distance_entry
          @Show_cmd
        Endif
      Endif
    Endif
    '
    If A=3 Or B=2
      T$=T$+"TORPEDOES "
      @Show_cmd
      Cstr$="at track number: "
      Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
      @Track_entry
      Ytext%=Crslin
    Endif
    Clr A,B,C,D
    @Cclear_middle
  Return
  '
  '                                   Engage sub-proc
  Procedure Llaunch
```

```
Mtxt$="   "+T$
Alert 2,Mtxt$+"...  ",0,"NUCLEAR CRUISE aircraft",A
If A=1
  T$=T$+"NUCLEAR "
  A=2
Endif
'
If A=2
  T$=T$+"CRUISE "
  @Show_cmd
  Cstr$="Enter number of missiles to fire (1-99): "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=1
  Hilim%=99
  Numlen%=2
  @Number_entry
  T$=T$+Pnum$
  Ytext%=Crslin
  Cstr$="Enter name of missile: "
  Print At(39-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)+" "
  @Cclear_middle
  @Show_cmd
  @Cruise_menu
  Lc1:          !## used to return from Cruise_menu_read
  Ytext%=8
  '
  If Bar$(Menu(0))=" BOL"
    T$=T$+"BOL BEARING "
    @Show_cmd
    @Bearing_entry
    T$=T$+" RANGE"
    Cstr2$="seeker on..."
    @Cclear_middle
    @Show_cmd
    Ytext%=8
    Print At(40-Int(Len(Cstr2$)/2),Ytext%);Cstr2$
    Ytext%=Ytext%+2
    @Distance_entry
    T$=T$+" RANGE"
    Cstr2$="seeker off..."
    @Cclear_middle
    @Show_cmd
    Ytext%=8
    Print At(40-Int(Len(Cstr2$)/2),Ytext%);Cstr2$
    Ytext%=Ytext%+2
    @Distance_entry
  Endif
  '
  If Bar$(Menu(0))=" PL2"
    T$=T$+"PL2 POSITION "
```

```
      @Llaunch_entry
    Endif
    '
    If Bar$(Menu(0))=" PL3"
      T$=T$+"PL3 POSITION "
      @Llaunch_entry
    Endif
    '
    If Bar$(Menu(0))=" PLTWO"
      T$=T$+"PLTWO POSITION "
      @Llaunch_entry
    Endif
    '
    If Bar$(Menu(0))=" PLTHREE "
      T$=T$+"PLTHREE POSITION "
      @Llaunch_entry
    Endif
    '
    If Bar$(Menu(0))=" TLAM"
      T$=T$+"TLAM AT "
      Cstr$="at which shorebase? "
      @Llaunch_entry_1
      @Llaunch_entry_2
    Endif
  Endif
  '
  If A=3
    @Ac_launch
  Endif
  '
  Clr A
  @Show_cmd
  @Force_menu_a
Return
'
'                                   ENGAGE sub-sub proc
Procedure Llaunch_entry
  @Show_cmd
  Cstr$="position..."
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  Print At(23,Ytext%);Lat_str$;
  @Lat_check
  Inc Ytext%
  Print At(23,Ytext%);Long_str$;
  @Long_check
  Inc Ytext%
  T$=T$+Upper$(Lat$)+" "+Upper$(Long$)+" "
  @Show_cmd
  @Cclear_middle
  Cstr$="Orientation: "
```

```
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  @Bearing_entry
  '
  @Show_cmd
  @Cclear_middle
  Cstr$=Cstr$+Pnum$+"   Semimajor axis: "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  @Distance_entry
  '
  @Show_cmd
  @Cclear_middle
  Cstr$=Cstr$+Pnum$+"   Semiminor axis: "
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  Inc Ytext%
  @Distance_entry
  '
  @Show_cmd
  @Cclear_middle
  Cstr$=Cstr$+Pnum$
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  @Llaunch_entry_2
Return
'
'                                    ENGAGE sub-sub proc
Procedure Llaunch_entry_1
  @Cclear_middle
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  @Show_cmd
  @Cclear_middle
Return
'
'                                     ENGAGE sub-sub proc
Procedure Llaunch_entry_2
  Alert 2," Waypoints desired? ",2,"  Yes   No",A
  N%=0
  If A=1
    @Cclear_middle
    Ytext%=4   !##  higher on screen than normal to allow many waypoints
    Waypt:
    Inc N%
    Inc Ytext%
    Print At(35,Ytext%);"Waypoint ";N%
    Inc Ytext%
    Print At(23,Ytext%);Lat_str$;
    @Lat_check
```

```
    Inc Ytext%
    Print At(23,Ytext%);Long_str$;
    @Long_check
    Inc Ytext%
    Mtxt$=" WAYPOINT "+Str$(N%)+" "+Upper$(Lat$)+" "+Upper$(Long$)
    T$=T$+Mtxt$
    Dec Ytext%
    Dec Ytext%
    Print At(20,Ytext%);Spc(40)
    Inc Ytext%
    Print At(20,Ytext%);Spc(40)
    Dec Ytext%
    Dec Ytext%
    Print At(30,Ytext%);Mtxt$
    Alert 2," Another waypoint? ",2,"  Yes   No",B
    If B=1
      Goto Waypt
    Endif
  Endif
  Clr A,B
Return
'
'                                    Take sub-menu
Procedure Ttake
  Mtxt$="   "+T$
  Alert 2,Mtxt$+"...  ",0,"track #  Base",A
  If A=1
    @Show_cmd
    Cstr$="Enter track number (ie, AB1234): "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If A=2
    T$=T$+"BASE "
    @Show_cmd
    Cstr$="Base name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Endif
  '
  @Show_cmd
  Mtxt$=" "+T$
  Alert 2,Mtxt$+" using...",0,"NUCLEAR CONVENTL either",B
  If B=1
    T$=T$+" NUCLEAR"
  Endif
  If B=2
    T$=T$+" CONVENTIONAL"
  Endif
  '
```

```
  @Cclear_middle
  @Tc_choice
  Clr A,B
  @Cclear_middle
Return
'
' ############ Second level Procedures for FORCE menu 'B' items ##############
'
'                                                AIRCRFT sub-menu
Procedure Ac_launch
  Menu Kill
  Aclaunch!=True  !## Set when Launch sequence is called; used to allow
  '           !## individual use of Load & Mission procs & A/C Command menu.
  '
  @Show_cmd
  Cstr$="How many aircraft? "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=0
  Hilim%=99
  Numlen%=2
  @Number_entry
  T$=T$+Pnum$
  '
  @Show_cmd
  @Cclear_middle
  Cstr$="Type of aircraft? "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
  '
  @Show_cmd
  @Cclear_middle
  Cstr$="Event name? "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)+" "
  '
  @Show_cmd
  Alert 2,"  Collective name...",0,"  Yes   No",A
  If A=1
    @Cclear_middle
    Cstr$="Collective name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
    '
    @Show_cmd
    Alert 2,"    select ...   ",0," LEADER  MEMBER",B
    If B=1
      T$=T$+" LEADER "
      A=2
    Endif
```

```
    If B=2
      T$=T$+" MEMBER"
      Inc Ytext%
      @Tc_choice
    Endif
  Endif
  '
  If A=2
    @Cclear_middle
    Inc Ytext%
    @Course_entry
    @Speed_entry
    T$=T$+" "
    @Altitude_entry
  Endif
  '
  Clr A,B
  '
  @Show_cmd
  @Ac_load
Return
'
'                                              AIRCRFT sub-menu item
Procedure Ac_load
  @Cclear_middle
  If Aclaunch!   !## ensures 'LOAD' occurs only once in T$
    Cstr$=" LOAD"
  Endif
  N%=0    !## set item counter
  Acload:
  Mtxt$="Enter the equipment LOAD    (up to 8 items)."
  Alert 2," "+Mtxt$,0,"  LOAD   End Load",A
  If A=1
    Mtxt$="LOAD how many of this item? (1-99): "
    Print At(39-Int(Len(Mtxt$)/2),Ytext%);Mtxt$;
    Lolim%=1
    Hilim%=99
    Numlen%=2
    @Number_entry
    Cstr$=Cstr$+" "+Pnum$
    '
    Mtxt$="                                                  "
    Print At(37-Int(Len(Mtxt$)/2),Ytext%);Mtxt$;
    Mtxt$="LOAD "+Pnum$+" (name of item?): "
    Print At(37-Int(Len(Mtxt$)/2),Ytext%);Mtxt$;
    Form Input 5,Fi$
    Cstr$=Cstr$+" "+Upper$(Fi$)
    Mtxt$="                                                  "
    Print At(37-Int(Len(Mtxt$)/2),Ytext%);Mtxt$;
    Dec Ytext%
    Dec Ytext%
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
```

```
    Inc Ytext%
    Inc Ytext%
    Inc N%
    If N%<8   !## allows only 8 different items
      Goto Acload
    Endif
  Endif
  '
  T$=T$+Cstr$+" "  !## Cstr$ is the Load command string
  @Show_cmd
  Clr A
  '
  If Aclaunch!   ! is True if Launch sequence is being used.
    @Mission_menu
  Endif
Return   !@Ac_load
'
'                                          AIRCRFT sub-menu item
Procedure Aalert
  @Show_cmd
  N%=1
  Cstr$="Status for what aircraft type? "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
  '
  Aalert:
  @Cclear_middle
  @Show_cmd
  If N%=1
    Cstr$="How many at 5 minute alert? "
  Endif
  If N%=2
    Cstr$="How many at 15 minute alert? "
  Endif
  If N%=3
    Cstr$="How many at 30 minute alert? "
  Endif
  '
  Inc N%
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=0
  Hilim%=99
  Numlen%=2
  @Number_entry
  T$=T$+" "+Pnum$
  If N%<4
    Goto Aalert
  Endif
  '
  Alert 2,"   Ordnance... ",2,"  Yes   No",A
  If A=1
```

```
    Cstr$="Enter ordnance (max 24 text characters): "
    Print At(32-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 24,Fi$
    T$=T$+" "+Upper$(Fi$)
  Endif
  Clr A
Return
'
'                                          AIRCRFT sub-menu item
Procedure Hhandover
  @Show_cmd
  Cstr$="Flight name: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
  @Cclear_middle
  @Show_cmd
  Cstr$="To... (force name): "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+" "+Upper$(Fi$)
Return
'
'                                          AIRCRFT sub-menu item
Procedure Oorbit
  @Show_cmd
  Cstr$="Enter radius (1-9999 nmi): "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+" "+Pnum$
  '
  For N%=1 To 2
    @Cclear_middle
    @Show_cmd
    Cstr$="Position "+Str$(N%)+" : "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
    Inc Ytext%
    Print At(23,Ytext%);Lat_str$;
    @Lat_check
    '
    Inc Ytext%
    Print At(23,Ytext%);Long_str$;
    @Long_check
    '
    T$=T$+" "+Upper$(Lat$)+" "+Upper$(Long$)
    Ytext%=Ytext%-2
  Next N%
  '
  @Cclear_middle
```

```
    @Show_cmd
    Alert 2,"        select ...       ",3,"SPEED TIME continue",A
    '
    If A=1
      T$=T$+" SPEED"
      @Speed_entry
      @Tc_choice
    Endif
    '
    If A=2
      @Time_entry
    Endif
    '
    Clr A
  Return
  '
  '                                          AIRCRFT Flt Cmds sub-menu item
  Procedure Aattach
    @Show_cmd
    Cstr$="...to collective flight (name?): "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
    '
    @Show_cmd
    Alert 2,"    select ...   ",0," LEADER  MEMBER",A
    '
    If A=1
      T$=T$+" LEADER"
    Endif
    '
    If A=2
      T$=T$+" MEMBER"
    Endif
    '
    @Cclear_middle
    @Tc_choice
    Clr A
  Return
  '
  '                                          AIRCRFT Flt Cmds sub-menu item
  Procedure Bbarrier
    @Show_cmd
    Cstr$="From position..."
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
    Inc Ytext%
    Inc Ytext%
    Print At(23,Ytext%);Lat_str$;
    @Lat_check
    '
    Inc Ytext%
    Print At(23,Ytext%);Long_str$;
```

```
    @Long_check
    '
    T$=T$+" "+Upper$(Lat$)+" "+Upper$(Long$)+" "
    '
    @Cclear_middle
    @Show_cmd
    @Bearing_entry
    Inc Ytext%
    @Distance_entry
    '
    @Cclear_middle
    @Show_cmd
    Cstr$="Using? "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+" "+Upper$(Fi$)
    '
    @Cclear_middle
    @Show_cmd
    Cstr$="Spacing? "
    Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
    Inc Ytext%
    Inc Ytext%
    @Distance_entry
    '
    Inc Ytext%
    @Tc_choice
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Cchaff
  @Show_cmd
  Cstr$="Barrier"
  Print At(40-Int(Len(Cstr$)/2),Ytext%);Cstr$
  Inc Ytext%
  @Minute_entry
  '
  Inc Ytext%
  @Tc_choice
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Ccover
  @Show_cmd
  Cstr$="Which track number? "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  @Track_entry
  '
  @Show_cmd
  Alert 2,"        select ...       ",3,"DISTANCE TIME continue",A
  '
  If A=1
```

```
    Inc Ytext%
    @Distance_entry
    @Tc_choice
  Endif
  '
  If A=2
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Ddeploy
  @Show_cmd
  Alert 2,"        select ...       ",0,"  BUOY   WIRE",A
  '
  If A=1
    T$=T$+"BUOY "
    @Show_cmd
    Cstr$="Buoy name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)
  Endif
  '
  If A=2
    T$=T$+"WIRE"
    @Show_cmd
    @Time_entry
  Endif
  '
  Clr A
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Rreconn
  @Show_cmd
  Mtxt$="   "+T$
  Alert 2,Mtxt$+"...  ",0,"track #  Base",A
  If A=1
    Cstr$="Enter track number (ie, AB1234): "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    @Track_entry
  Endif
  '
  If A=2
    T$=T$+"BASE "
    @Show_cmd
    Cstr$="Base name: "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
```

```
    T$=T$+Upper$(Fi$)
    @Show_cmd
    @Time_entry
  Endif
  Clr A
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Rrefuel
  @Show_cmd
  Cstr$="Refueler name: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  @Show_cmd
  @Time_entry
Return
'
'                                          AIRCRFT Flt Cmds sub-menu item
Procedure Tturn
  @Show_cmd
  @Course_entry
  @Show_cmd
  @Time_entry
Return
'
'                                    AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Aair
  @Show_cmd
  Cstr$="REPORT Air tracks every..."
  Print At(38-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  @Minute_entry
  Inc Ytext%
  @Tc_choice
Return
'
'                                    AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Eesm
  @Show_cmd
  Cstr$="REPORT ESM tracks every..."
  Print At(38-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  @Minute_entry
  Inc Ytext%
  @Tc_choice
Return
'
'                                   AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Oon
  @Show_cmd
  Cstr$="REPORT On circuit number"
  Cstr2$=Cstr$+" : "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr2$;
```

```
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+Pnum$
  @Show_cmd
  '
  Cstr$="REPORT On circuit number "+Pnum$+" or : "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Lolim%=1
  Hilim%=9999
  Numlen%=4
  @Number_entry
  T$=T$+" "+Pnum$
  @Show_cmd
  '
  Alert 2,"   Violate EMCON? ",0,"  Yes   No",A
  If A=1
    T$=T$+" YES"
  Endif
  If A=2
    T$=T$+" NO"
  Endif
  Clr A
  '
  Inc Ytext%
  @Tc_choice
Return
'
'                                  AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Pposition
  @Show_cmd
  Cstr$="REPORT Position & logistics every..."
  Print At(41-Int(Len(Cstr$)/2),Ytext%);Cstr$
  @Minute_entry
  Inc Ytext%
  @Tc_choice
Return
'
'                                  AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Ssurface
  @Show_cmd
  Cstr$="REPORT Surface tracks every..."
  Print At(41-Int(Len(Cstr$)/2\,Ytext%);Cstr$
  @Minute_entry
Return
'
'                                  AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Tttime
  @Show_cmd
  Print At(26,Ytext%);"Enter start minute (1-999): ";
  Lolim%=1
```

```
  Hilim%=999
  Numlen%=3
  @Number_entry
  T$=T$+"TIME "+Pnum$
Return
'
'                                   AIRCRFT Flt Cmds REPORT sub-menu item
Procedure Uusing
  @Show_cmd
  Cstr$="REPORT Using what policy?: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  Inc Ytext%
  @Tc_choice
Return
'
'                                                       COMMS sub-menu item
Procedure Ccommtext
  @Show_cmd
  Cstr$="Using what path?: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)+" "
  '
  @Show_cmd
  @Cclear_middle
  Cstr$="To which receiver?: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)+" "
  '
  Ccomtxt:
  @Show_cmd
  @Cclear_middle
  Alert 2,"  Another receiver? ",0,"  Yes   No",A
  If A=1 Then
    Cstr$="Enter receiver name. "
    Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
    Form Input 5,Fi$
    T$=T$+Upper$(Fi$)+" "
    Goto Ccomtxt
  Endif
  Clr A
  I%=8
  @Text_entry   !## common Comms text entry
Return
'
'                                                       COMMS sub-menu item
Procedure Eembark
  @Show_cmd
  Cstr$="Enter force name: "
```

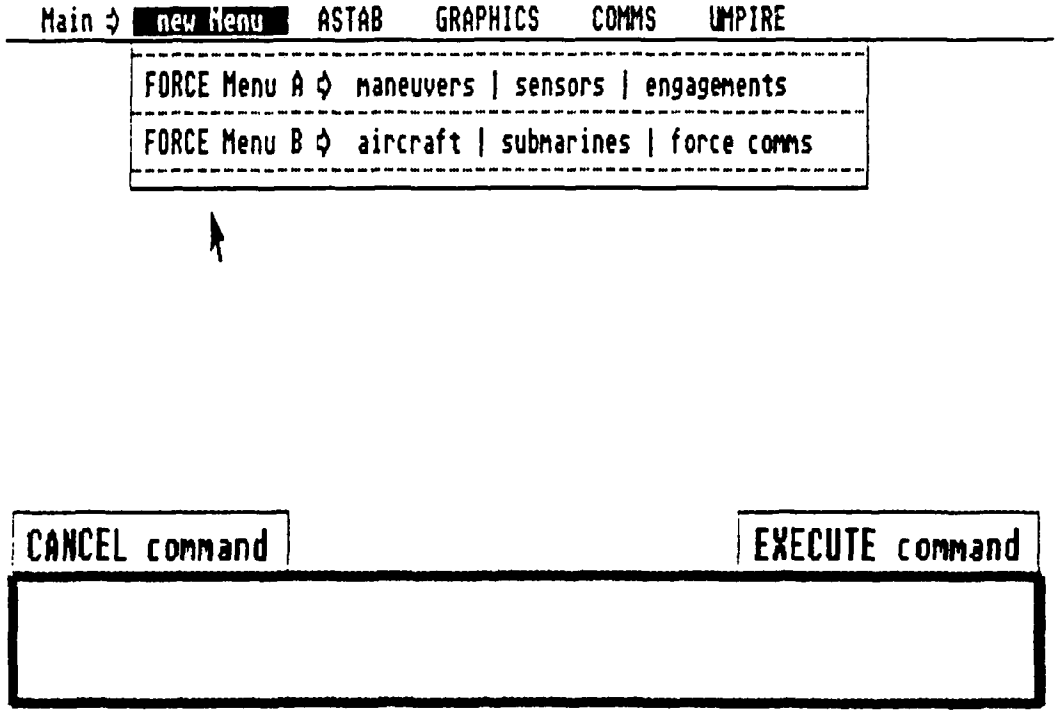
```
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
  '
  @Show_cmd
  Alert 2,"    On what platform?   ",0," Orange  Blue continue",A
  If A=1 Then
    T$=T$+" ORANGE "
  Endif
  If A=2 Then
    T$=T$+" BLUE "
  Endif
  Clr A
  '
  @Show_cmd
  @Cclear_middle
  Cstr$="Enter view number: "
  Print At(37-Int(Len(Cstr$)/2),Ytext%);Cstr$;
  Form Input 5,Fi$
  T$=T$+Upper$(Fi$)
Return
'
' ##############################################################################
'                                END of Program
' ##############################################################################
```

APPENDIX B

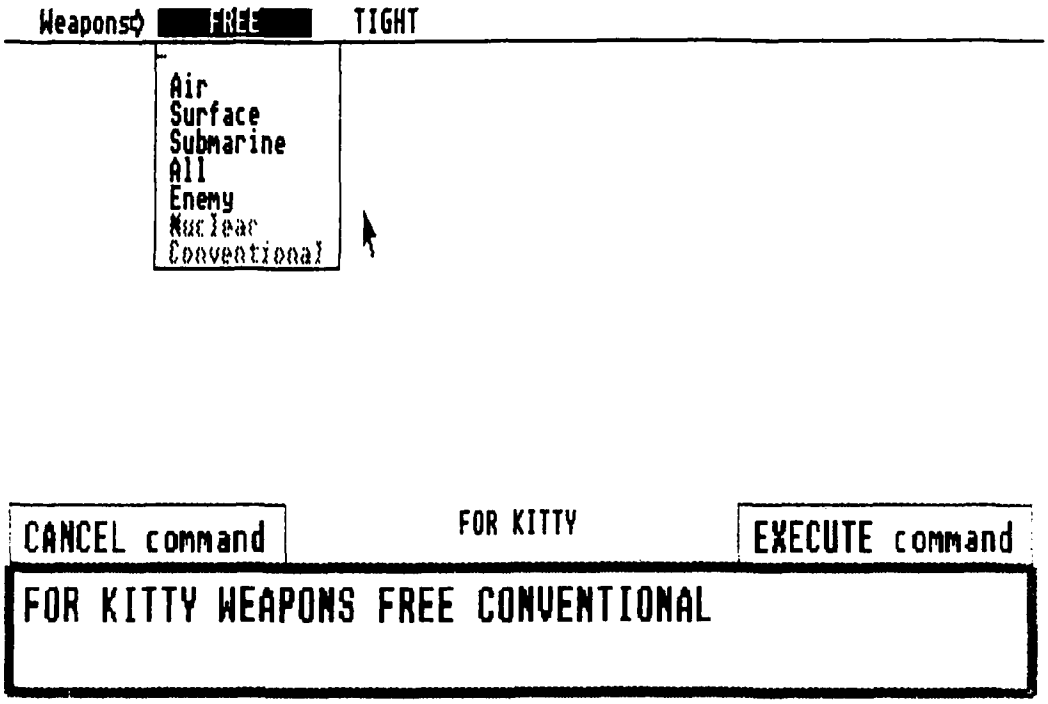
RESA Interface Program User's Guide

This appendix is a simple guide to facilitate use of the RESA Interface Program (RIP). To effectively use this guide, some familiarity with RESA is necessary, as no attempt to define RESA commands is made herein.

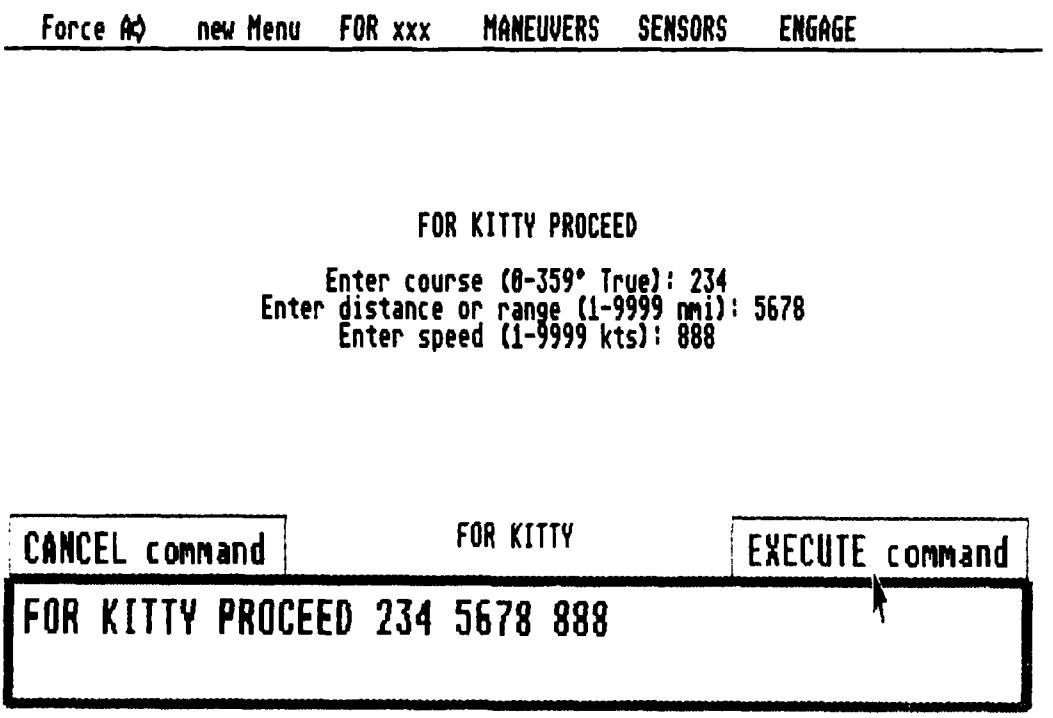
The starting screen display will provide the Main Menu, one of the three primary control menus. The other two are: Force Menu A and Force Menu B. Access to each primary menu is available from each other primary menu using the "new Menu" menu bar selection.



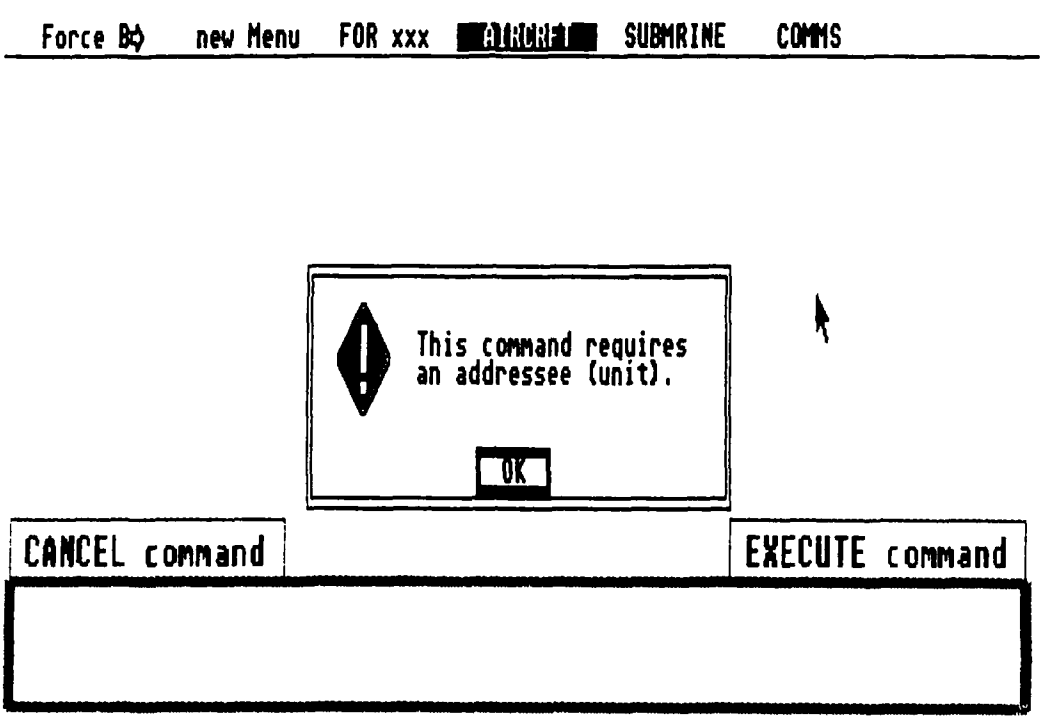
Orders are "built" by successively selecting menu headings and using the "mouse" to select desired commands. As commands are selected, the RIP may ask for additional data or second-level commands. Appropriate secondary/tertiary menus will be displayed as needed.



Make a specific choice from the screen or a secondary menu, or to use the keyboard to enter alpha-numeric characters. Considerable error-checking is performed if the keyboard is used.



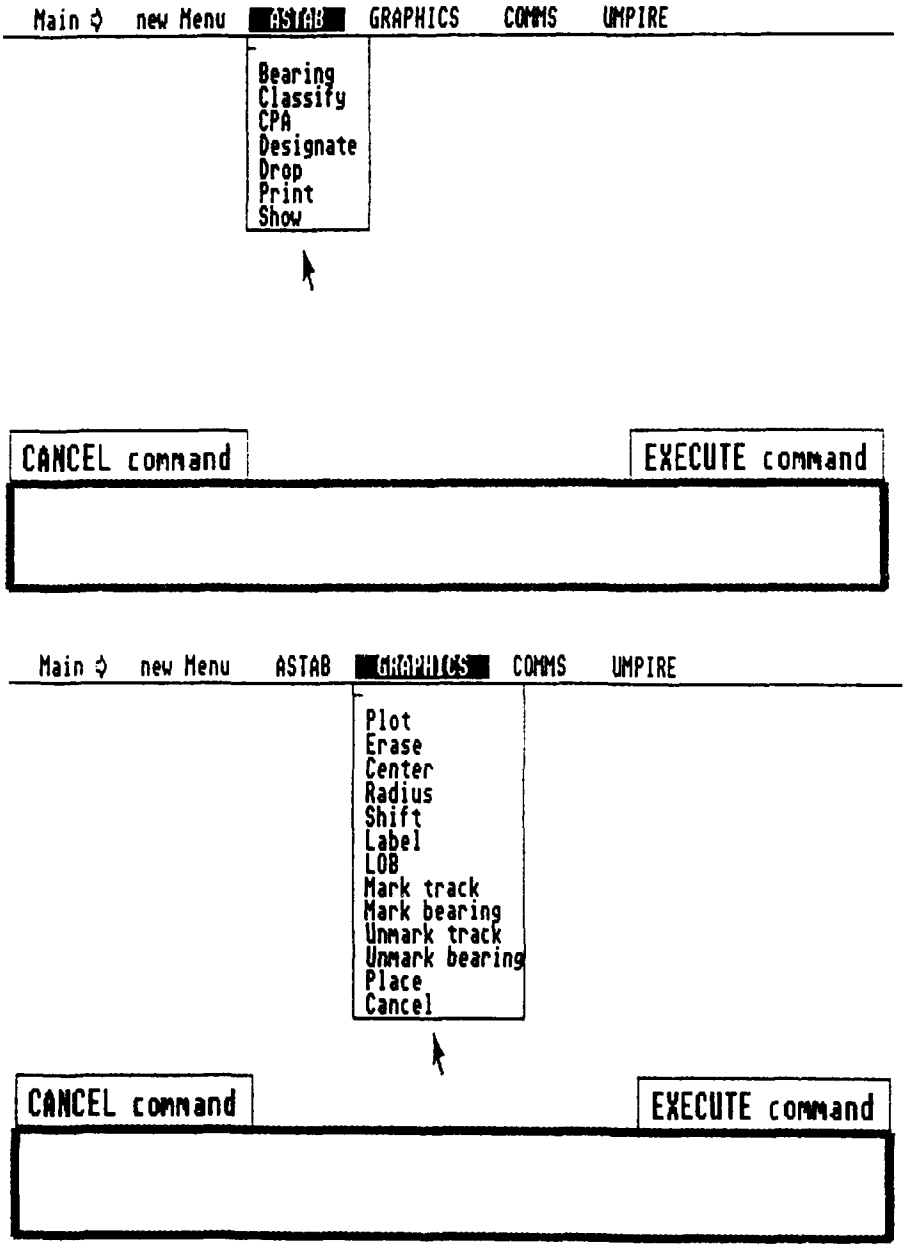
If a command is supposed to be preceded by a force addressee, an "alert box" will appear and request it.

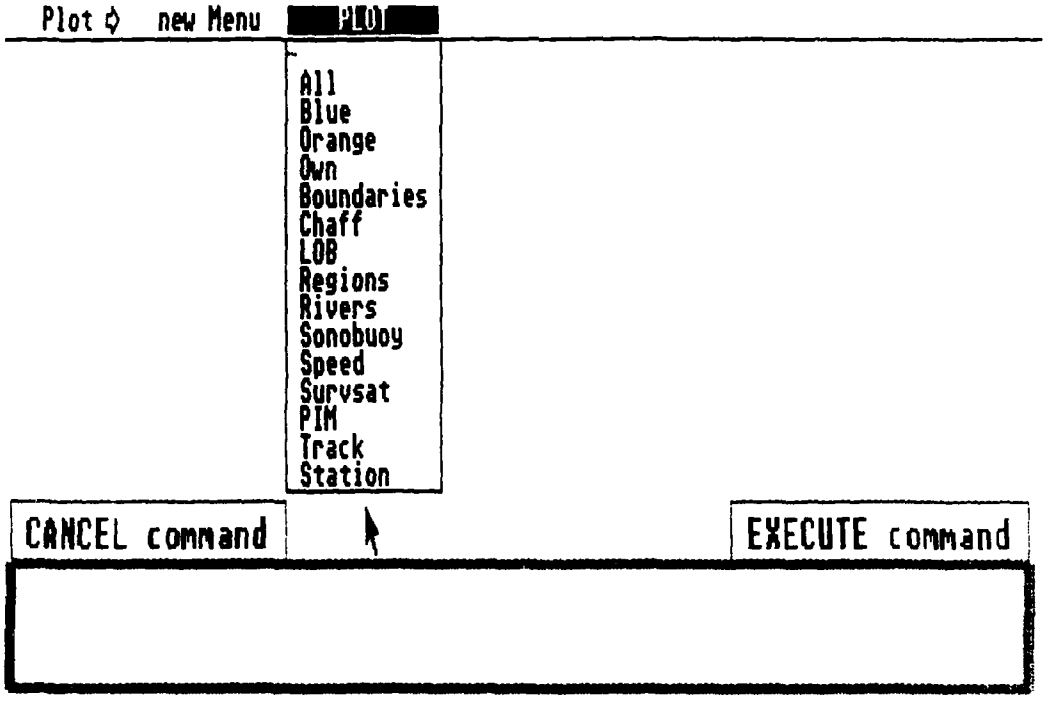


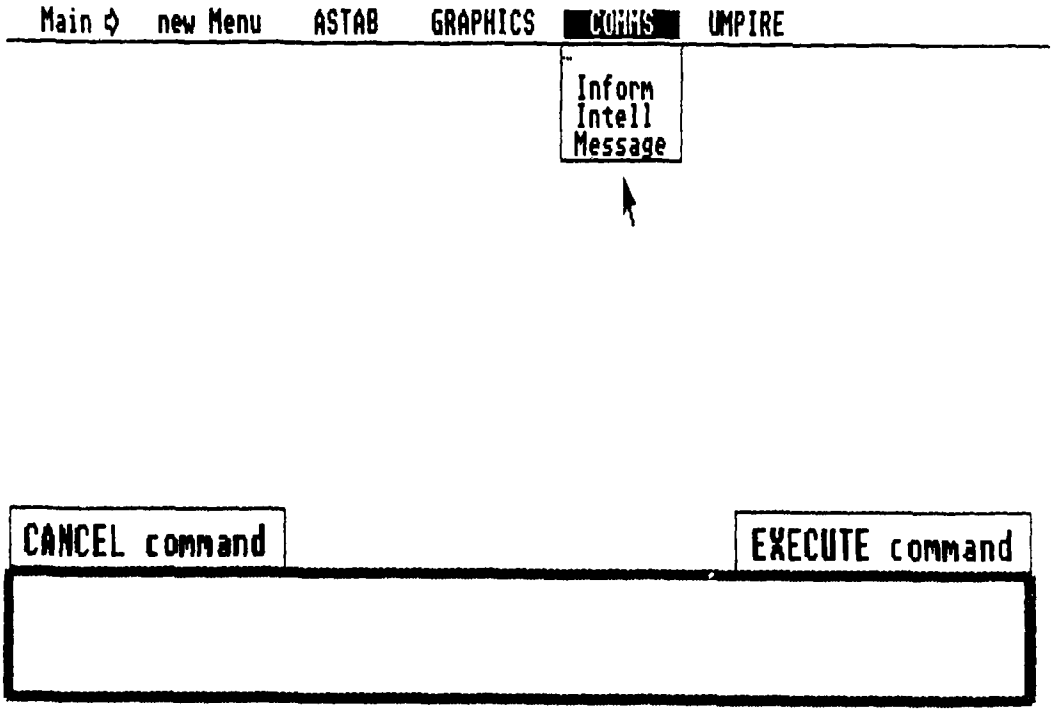
As they are being built, orders are displayed in a command box at the bottom of the screen. When an order is syntactically correct, the user is given the choice to Execute or Cancel it. Execution or Cancellation of non-completed orders is not allowed.

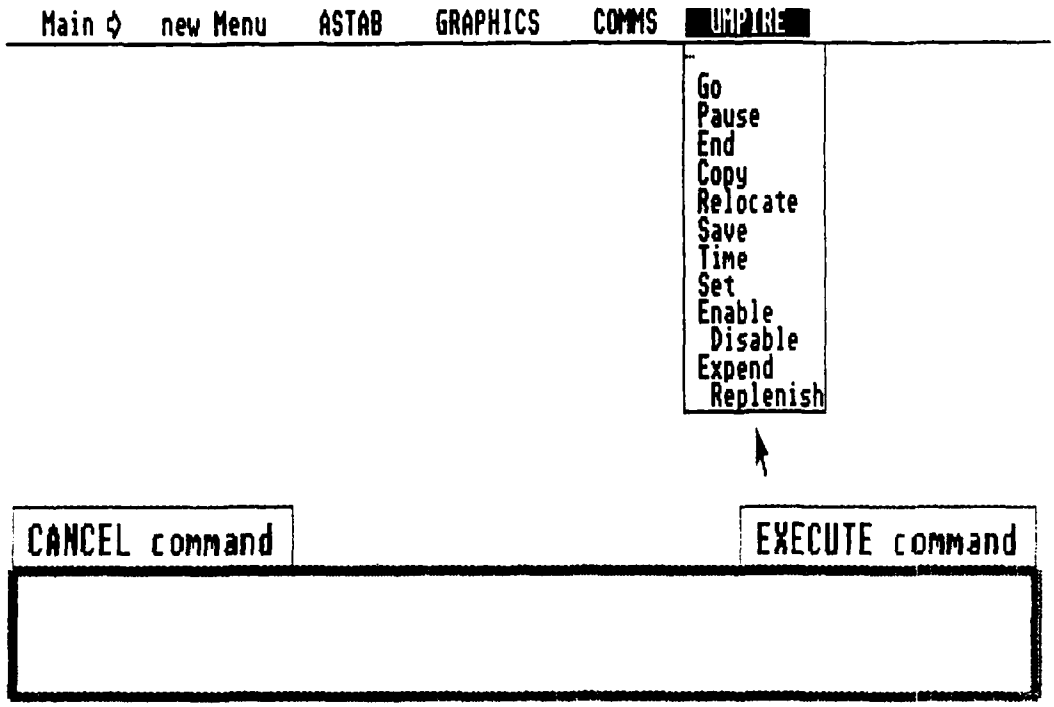
The following are examples of what commands are displayed whenever certain menu bar headings are selected.

I. MAIN Menu Selections

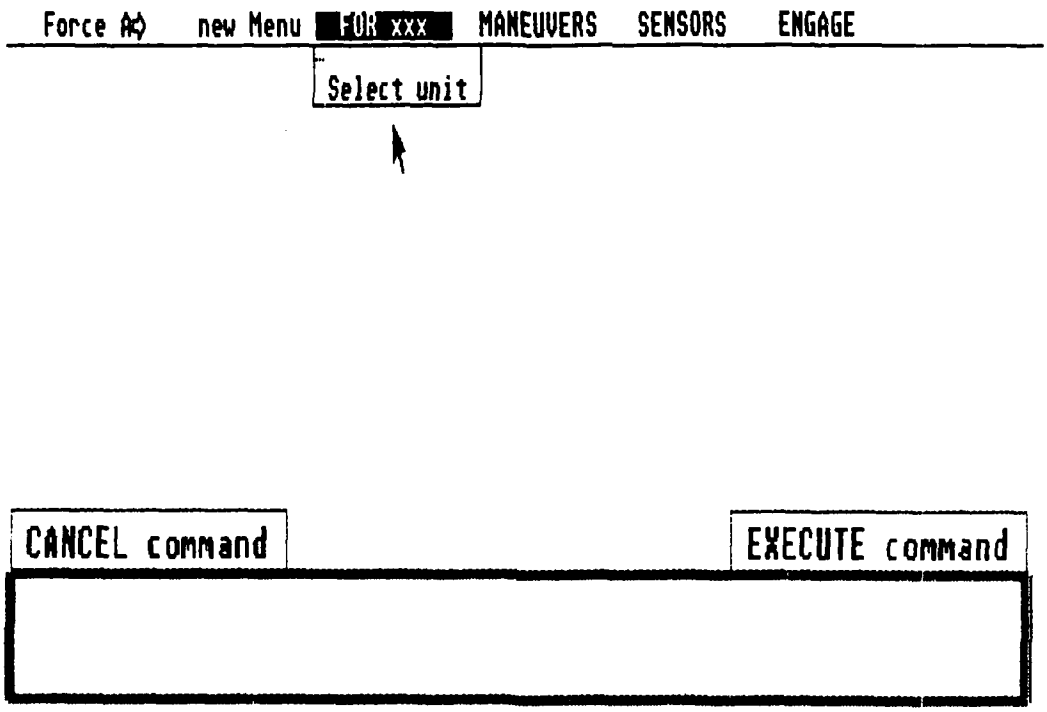


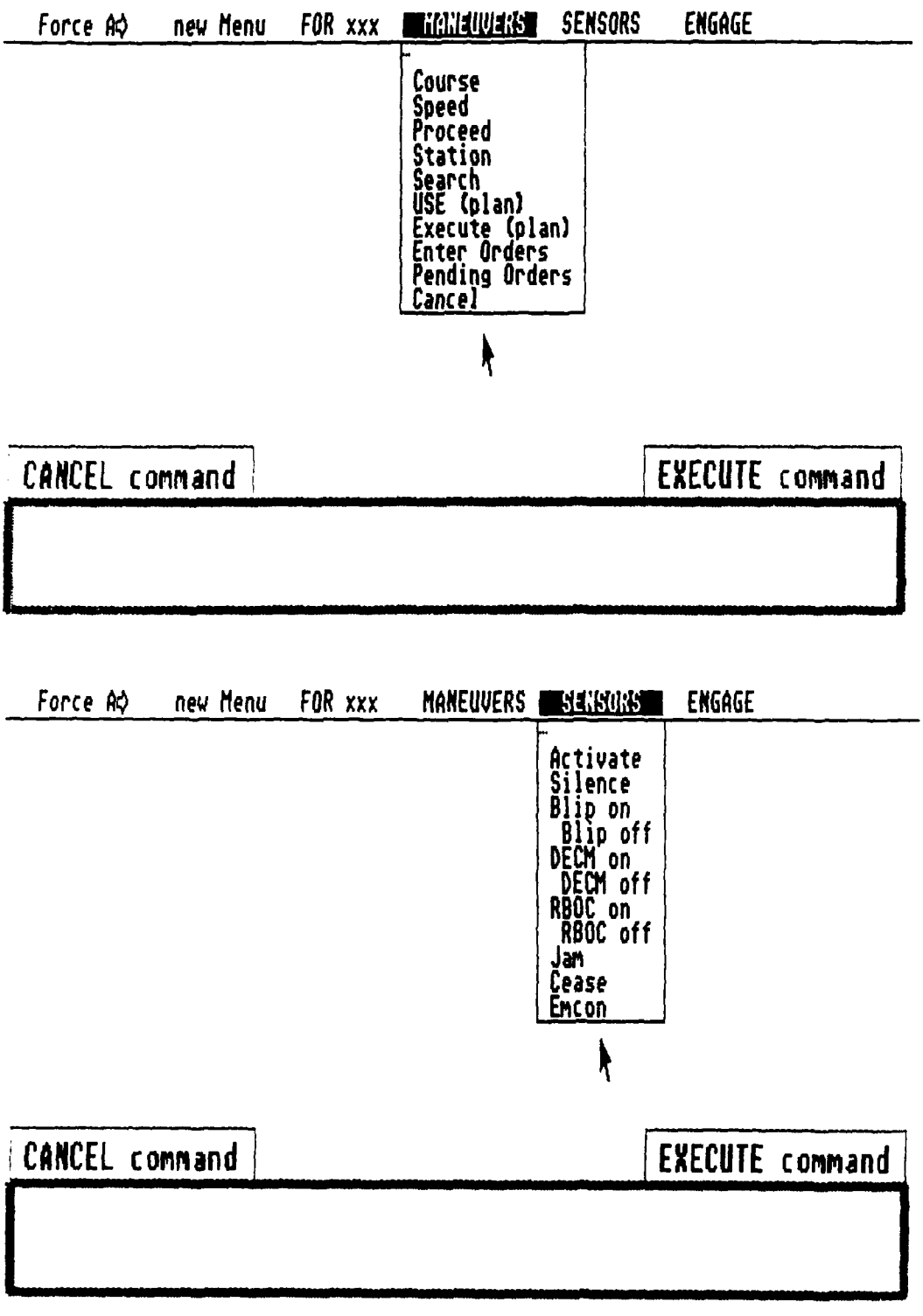

Plot ◊ new Menu PLOT
All
Blue
Orange
Own
Boundaries
Chaff
LOB
Regions
Rivers
Sonobuoy
Speed
Survsat
PIM
Track
Station
CANCEL command
EXECUTE command

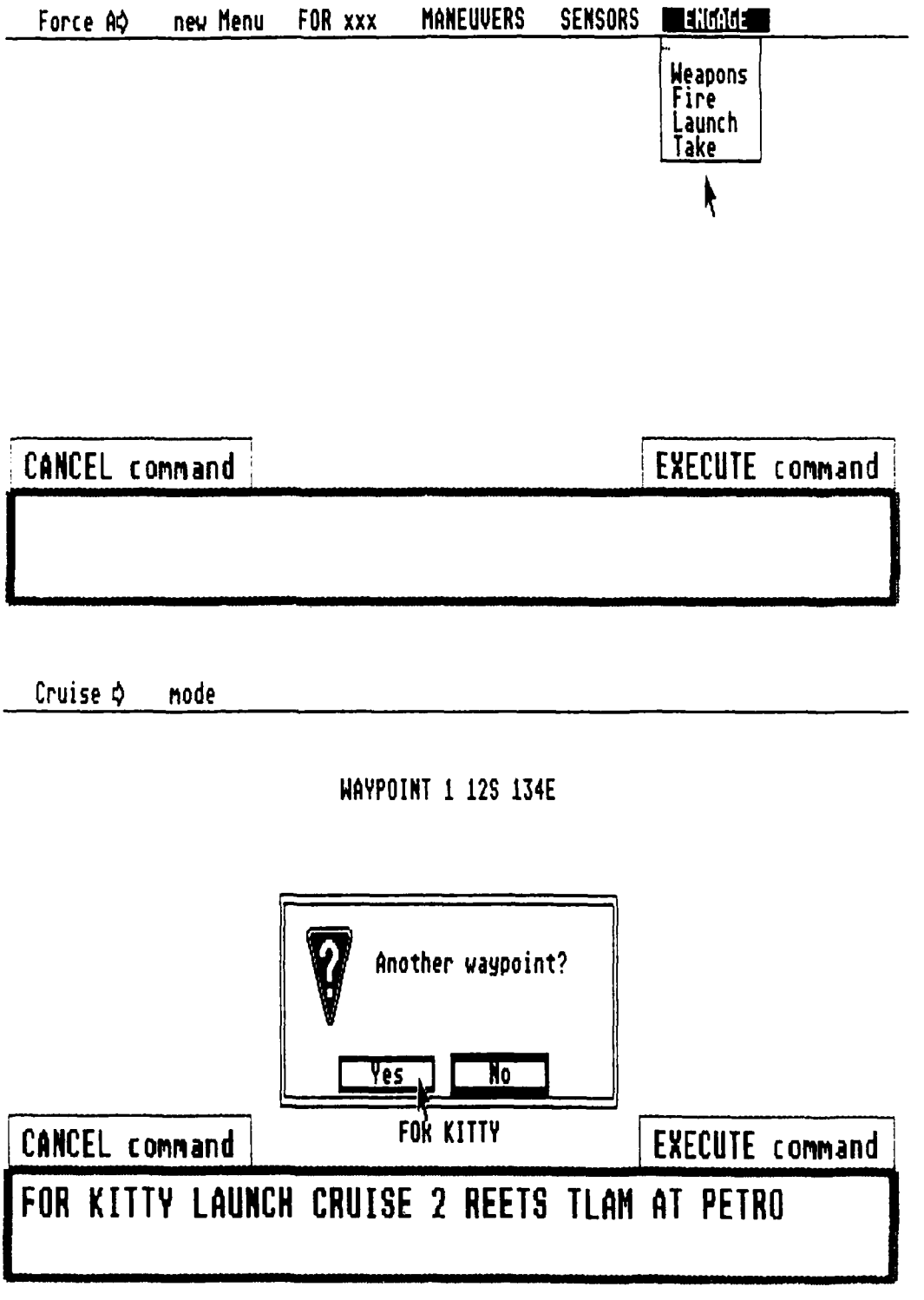

Main ◊ new Menu ASTAB GRAPHICS WORMS UMPIRE
Inform
Intell
Message
CANCEL command
EXECUTE command


Main ◊
new Menu
ASTAB
GRAPHICS
COMMS
UNPARE
Go
Pause
End
Copy
Relocate
Save
Time
Set
Enable
Disable
Expend
Replenish
CANCEL command
EXECUTE command

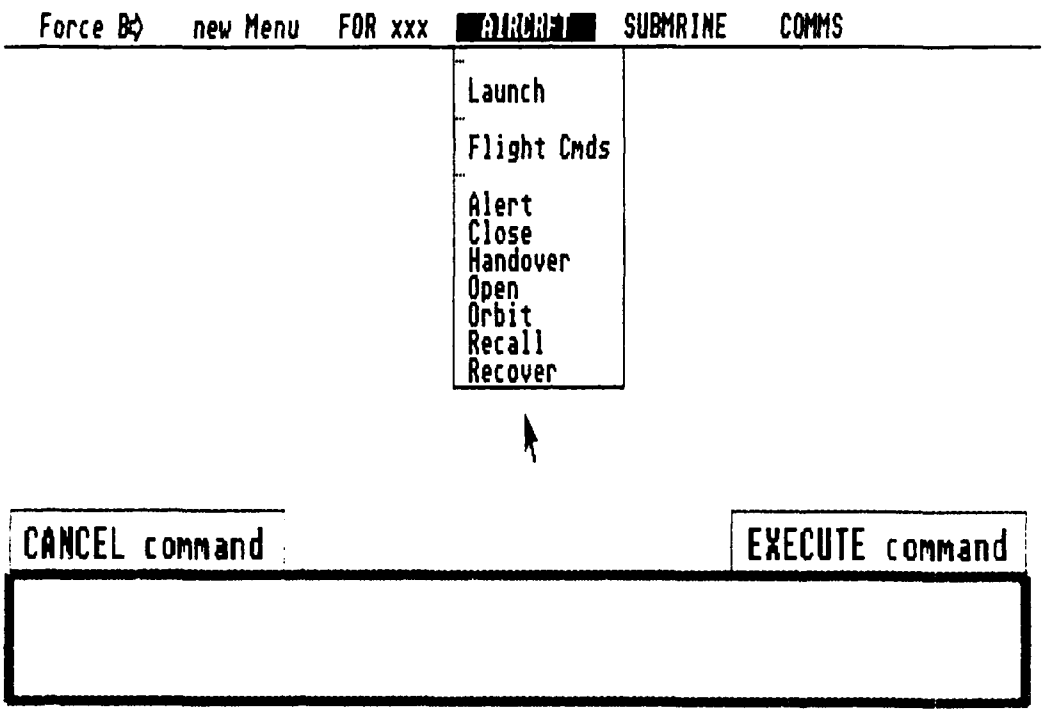
II. FORCE Menu A Selections




Force A◊ new Menu FOR xxx MANEUVERS SENSORS ENGAGE
Course
Speed
Proceed
Station
Search
USE (plan)
Execute (plan)
Enter Orders
Pending Orders
Cancel
CANCEL command
EXECUTE command
Force A◊ new Menu FOR xxx MANEUVERS SENSORS ENGAGE
Activate
Silence
Blip on
Blip off
DECM on
DECM off
RBOC on
RBOC off
Jam
Cease
Emcon
CANCEL command
EXECUTE command

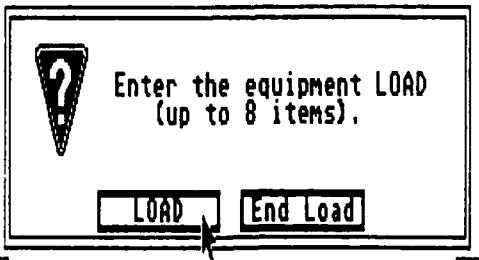

Force A◊
new Menu
FOR xxx
MANEUVERS
SENSORS
ENGAGE
Weapons
Fire
Launch
Take
CANCEL command
EXECUTE command
Cruise ◊
mode
WAYPOINT 1 12S 134E
Another waypoint?
Yes
No
FOR KITTY
CANCEL command
EXECUTE command
FOR KITTY LAUNCH CRUISE 2 REETS TLAM AT PETRO

III. FORCE Menu B Selections



Force B◊ new Menu FOR xxx [redacted] SUBMRINE COMMS

LOAD 12 HRPON

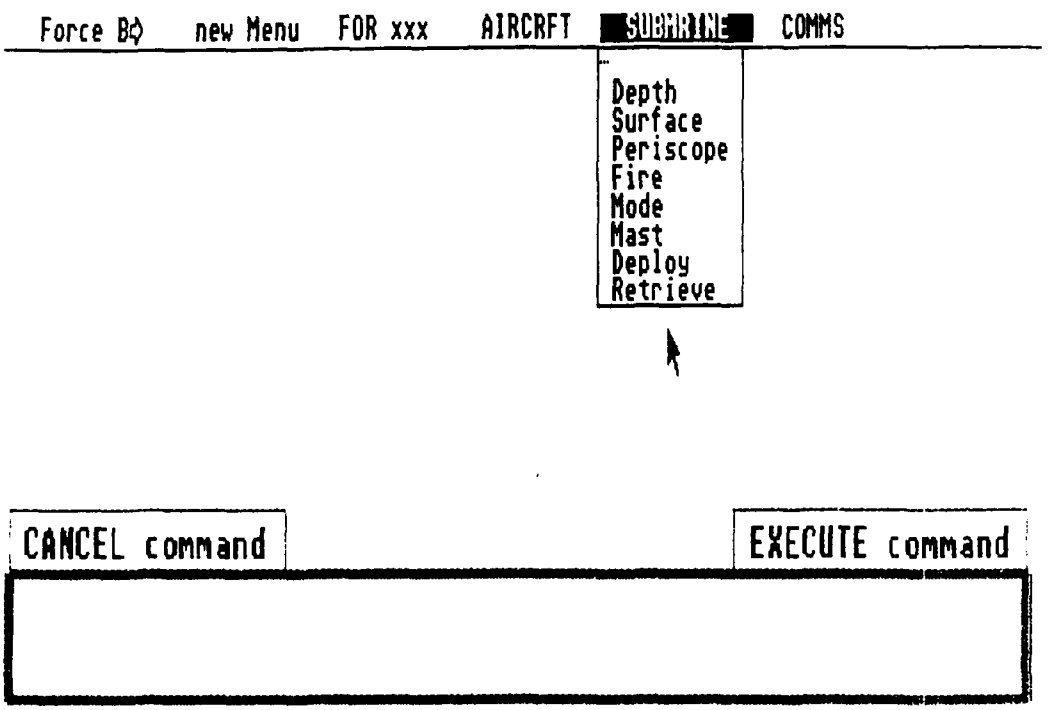


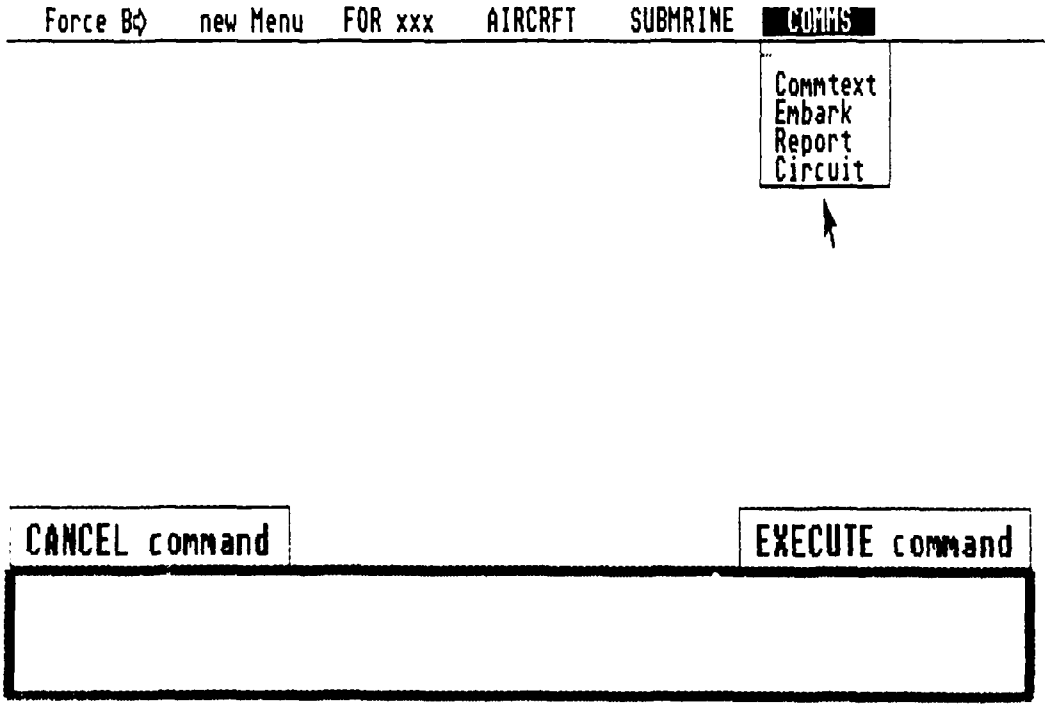
CANCEL command | FOR KITTY | EXECUTE command

FOR KITTY LAUNCH 12 F14A TEAL 345 450 20000

Flt Cmd◊ [redacted] D - R S - Z

CANCEL command | FOR KITTY | EXECUTE command


Force B◊ new Menu FOR xxx AIRCRFT SUBMRINE COMMS
Depth
Surface
Periscope
Fire
Mode
Mast
Deploy
Retrieve
CANCEL command
EXECUTE command


Force B◊ new Menu FOR xxx AIRCRFT SUBMRINE COMMS
Commtext
Embark
Report
Circuit
CANCEL command
EXECUTE command

BIBLIOGRAPHY

Adams, R.M., A Software Architecture for a Commander's Display System, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1987.

Balma, Phillip and William Fitler. Programmer's Guide to GEM. Berkeley: Sybex, 1986.

Copeland, D.M., The Development of a User-friendly Interface for the Battle Group Tactical Trainer: A Prototyping Approach, M.S. Thesis, Naval Postgraduate School, Monterey, California, September 1987.

Gerits, K., L. Englisch and R. Bruckmann. ATARI ST Internals. Michigan: Abacus, 1985.

Hilchner, Hendrika GFA BASIC. Michigan: Michtron, 1986.

Irving, R., MacNWISS: Using the Macintosh as a Command Input Terminal for NWISS, Project, Naval Postgraduate School, Monterey, California, 1986.

LeFever, M.A., Speech Recognition in a Command and Control Workstation Environment, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1987.

Lower, S.L., The Development of Visual Interface Enhancements for Player Input to the JTLS Wargame, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1987.

Manson, R.B., and Michael E. Wright, Comparison of Continuous Speech, and Keyboard Input to an Interactive Warfare Simulation in Various C3 Environments, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1985.

Ostrowski, Frank. GFA BASIC Book. Michigan: Michtron, 1987.

Ostrowski, Frank. GFA Compiler. Michigan: Michtron, 1986.

Stevens, N.G., The Application of Current User Interactive Technology to Interactive Wargaming Systems, M.S. Thesis, Naval Postgraduate School, Monterey, California, September 1987.

Sweeney, M.J. and K.J. Bitar, An Analysis of Friendly Input Devices for the Control of the Naval Warfare Interactive Simulation System, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1987.

Szczepanowski, Norbert and Bernd Gunther. ATARI ST GEM Programmer's Reference. Michigan: Abacus, 1985.

INITIAL DISTRIBUTION LIST

		No. copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Commander, Naval Space Command Attn: Code N3 Dahlgren, Virginia 22448	1
4.	Commander, United States Space Command Attn: Technical Library Peterson AFB, Colorado 80914	1
5.	Navy Space System Division Chief of Naval Operations (OP-943) Washington, D.C. 20305-2000	1
6.	CDR J. Stewart, Code 55st Naval Postgraduate School Monterey, California 93943-5000	2
7.	Naval Ocean Systems Center RESA Facility San Diego, California 92152	1
8.	Curricular Office, Code 39 Naval Postgraduate School Monterey, California 93943-5000	2
9.	OJCS / J-8 attn: LTCOL Duff The PENTAGON Washington, D.C. 20419	1
10.	ATARI Corporation 1196 Borregas Ave Sunnyvale, California 94086	1
11.	LCDR G. L. Yungk NSG Command Detachment, Potomac Washington, D.C. 20374-5036	1